Modeling and Computational Issues in the Development of Batch Processes

by

Russell John Allgor

Submitted to the Department of Chemical Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Chemical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 1997

Author . . . .
Department of Chemical Engineering
June 16, 1997

Certified by . . . .
Paul I. Barton
Assistant Professor
Thesis Supervisor

Certified by . . . .
Lawrence B. Evans
Adjunct Professor
Thesis Supervisor

Accepted by . . . .
Robert Cohen
St. Laurent Professor of Chemical Engineering
Chairman, Committee on Graduate Students

Modeling and Computational Issues in the Development of Batch Processes

by

Russell John Allgor

Submitted to the Department of Chemical Engineering
on June 16, 1997, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Chemical Engineering

Abstract

The rapid development of an efficient process to manufacture a new or modified product within an existing batch manufacturing facility is critical to the success of many specialty chemical and synthetic pharmaceutical companies. This thesis employs process modeling technology as the basis for an integrated batch process development methodology that complements and enhances laboratory and pilot scale experimentation. Examples demonstrate that significant benefits can be realized for these industries.

To develop optimal batch processes using detailed mathematical models, the continuous decisions defining the operating policies of the processing tasks and the discrete decisions defining the process structure and allocation of plant resources must be made simultaneously. The first rigorous decomposition algorithm that simultaneously considers both types of decisions is derived; the algorithm also extends to general mixed time invariant integer dynamic optimization problems. This decomposition algorithm requires subproblems that yield rigorous upper and lower bounds on the objective, and robust numerical techniques to solve each subproblem. *Screening models* are derived to provide rigorous lower bounds on the manufacturing cost; upper bounds on the cost are provided by the solution of a dynamic optimization problem. The robustness, accuracy, and efficiency of the numerical solution algorithms for the simulation and optimization of detailed discrete/continuous dynamic models is also improved, allowing the solution of the dynamic optimization subproblem to be performed more reliably.

Screening models exploit domain specific knowledge to obtain rigorous lower bounds on the manufacturing cost. The lower bounding property of the screening models is proven for networks of reaction and distillation tasks and demonstrated on several case studies that illustrate the ability of the screening models to handle aspects of process synthesis. The design targets provided by the solution of these models facilitate rapid decision making during the early stages of process development, enhance the application of other design methodologies, and facilitate the formulation and solution of the dynamic optimization subproblems required within the decomposition algorithm.

Sophisticated equation based modeling environments provide modeling flexibility by decoupling the solution procedures from the model definition but, at the same time, place severe expectations on the numerical integration techniques. The application of these environments to the simulation and optimization of batch reaction and distillation tasks uncovers several previously unreported numerical problems. This thesis proves that the observed numerical difficulties are caused by an ill-conditioned corrector iteration matrix, demonstrates that the accuracy of DAE integration codes is limited by the condition number of the corrector iteration matrix, and explains how the integration code's error control strategy can permit the generation of 'spikes'. Automated scaling techniques are developed and implemented to permit the efficient solution of poorly scaled problems and to mitigate the effects of ill-conditioned models; it is proven that this scaling comes very close to the optimal scaling for the sparse unstructured matrices with which we are concerned. In addition, a novel strategy is developed to start DAE integration codes efficiently at the frequent discontinuities experienced in such simulations and optimizations.

The advantages of this integrated design methodology are demonstrated through a series of realistic examples exhibiting the complexity of typical industrial applications.

Thesis Supervisor: Paul I. Barton
Title: Assistant Professor

Thesis Supervisor: Lawrence B. Evans
Title: Adjunct Professor

Dedicated to my family for their love, encouragement, and support.

“Given the pace of technology, I propose we leave math to the machines and go play outside.”

— Calvin, Calvin and Hobbes by Bill Waterson (1992)

Acknowledgments

I would like to thank Professor Paul Barton for providing guidance and encouragement during the course of my thesis research. I have learned a lot from the discussions we have had over the years; I truly appreciate the attention you have given my thesis during the past few months. You have been a good advisor and a true friend. I would also like to thank Professor Larry Evans for the inspiration driving this research and for convincing me to come to MIT.

I would like to thank the members of my research group and friends for providing me with the opportunity to discuss ideas and broaden my knowledge through fruitful discussions. Howard, John, and Santos have taken time to discuss the numerical aspects of my research, allowing me to iron out the details of this portion of my research. Wade and John have allowed me to spend more time on my research by taking over the management of the computer systems and ABACUSS. Thanks to Bill for fixing bugs that I have run into in a timely fashion. Taeshin's generosity with his afternoon snacks has kept me from going hungry over the past few months, and Christophe kept me up to date with the sports news from Europe. Thanks to Mingjuan and Kamel for helping me with portions of my research, and to Berit for making sure that I have had a research group through all these years at MIT.

I would like to thank Elaine for organizing the recruiting in the department. She went out of her way to help me out after my surgery and provided me with opportunities that may have otherwise been wasted.

I really appreciate all the support my friends have given me over the past few years. Without their support I would not have been able to get this far. The soccer team at MIT has given me some great friends and really helped me to enjoy the time I have spent here. Special thanks to Walter, Josh, and Steffen who have kept the team organized and made the game a lot of fun. I'm also glad that Alex decided to go to HBS for school. It was great to have a good friend in the area who was not associated with MIT.

Most of all, I would like to thank my family. The past year has been extremely difficult for me, and the love and support you have given me has helped me get through the thesis, recover from back surgery, and deal with everything else that has happened. Mom and Dad, you have come to my aid when I needed your help. Alison, your visits to Boston were a great help. I know I can always count on your support.

In conclusion, I would like to thank the US Department of Energy for financial support.

Contents

1 Introduction **21**

1.1 Batch Process Manufacturing 23

1.2 Batch Process Development 25

1.3 Design Methods for Batch Process Development 27

1.4 Screening Models for Batch Process Development 33

1.5 Rigorous Decomposition Algorithm 34

1.6 Numerical Issues in the Detailed Simulation of Batch Processes . . . 37

1.7 Outline of Thesis 42

2 Batch Process Development **45**

2.1 Previous Research 46

2.1.1 Design with Fixed Recipes 48

2.1.2 Design with Recipe Modifications 51

2.1.3 Coupling the Structure and Performance Subproblems 57

2.2 Applying Screening Models to Process Development 59

2.3 Scope of Development Problems Considered 64

2.4 Decomposition Algorithm for Batch Process Development 70

2.5 Summary 75

3 Screening Models for Batch Process Development **77**

3.1 Deriving Screening Models for Reaction/Distillation Networks 78

3.1.1 Process Abstraction 79

3.1.2 Batch Distillation Composition Bounds 81

3.1.3 Reactor Targeting Model 87

3.2 Time Averaged Material Balances 88

3.3 Bounding Distillation Processing Time and Utility Requirements . . . 92

3.3.1 Distillation Processing Time Bounds 93

3.3.2 Bounding the Distillation Utility Requirements 96

3.3.3 Definition of Bottoms Cuts 100

3.4 Equipment Allocation 101

3.5 Process Performance and Production Cost 104

3.6 Formulating the model to be solved 107

3.7 Conclusions 108

3.8 Notation 110

3.8.1 Indexed Sets 110

3.8.2 Integer Variables 110
3.8.3 Binary Variables 110
3.8.4 Exact linearizations of bilinear products of binary variables . . 111
3.8.5 Continuous Variables 111
3.8.6 Parameters 113

4 Using Screening Models to Identify Favorable Processing Structures 115

4.1 Process Description 116
4.2 Design Constraints 117
4.3 Reaction targets 120
4.3.1 Bounding the selectivity and extent of reaction 121
4.3.2 Convexifying the Extent/Time Boundaries 124
4.3.3 Minimum Extents of Reaction 132
4.4 Process Superstructure 135
4.5 Solutions of the Screening Models 136
4.5.1 Solution obtained from the First Superstructure 137
4.5.2 Solution obtained from the Second Superstructure 141
4.5.3 Solution Comparison 145
4.6 Computational Considerations 146
4.6.1 Size of the Models solved 146
4.6.2 Scaling of the Linear Programs 147
4.6.3 Solution Procedure 148
4.6.4 Linearization of Bilinear Terms 151
4.6.5 Influencing the Branch and Bound Algorithm 155
4.6.6 Tailored Solution Procedures 156
4.6.7 Representation of Batch Distillation Boundaries 157
4.7 Summary 157
4.8 Notation 158
4.8.1 Indexed Sets 158
4.8.2 Binary Variables 159
4.8.3 Variables 159
4.8.4 Parameters 159

5 Siloxane Monomer Case Study 161

5.1 Laboratory Scale Process 162
5.1.1 First Reaction Task 162
5.1.2 Second Reaction Task 163
5.1.3 Third Reaction Task 164
5.1.4 Design Constraints 165
5.2 Case Study I: Comparison of minimum cost versus minimum waste . 166
5.2.1 Solution 169
5.3 Case Study II: Including Reaction Targets 177
5.3.1 First Reaction Task 177
5.3.2 Solutions to Case Study II 186
5.3.3 Case III: Disposing of Recycle Streams 195

5.4 Conclusions 196

6 Numerical Issues in the Simulation and Optimization of Hybrid Dynamic Systems 201

6.1 Accuracy of Solution Procedures 202

6.1.1 Backward Error and Conditioning 206

6.2 Efficiency of Integration Codes 208

6.3 Mathematical Background 209

6.3.1 BDF Integration Codes 209

6.3.2 Dynamic Optimization 213

6.3.3 Rounding Error Analysis 220

6.3.4 Scaling of Linear Systems 225

6.3.5 Row Equilibration 226

6.3.6 Properties of Newton's Method 229

6.4 Summary 233

7 Automatic Scaling of Differential-Algebraic Systems 235

7.0.1 Modeling Flexibility Derived from the Automatic Scaling of DAE Models 236

7.1 Demonstration of Problem 237

7.2 Explanation of the Phenomenon 242

7.2.1 Generation of a 'spike' 243

7.2.2 Truncation Error Criterion 246

7.3 Ill-conditioned Corrector Iterations 248

7.4 Stiffness, Conditioning, and Index 251

7.4.1 Stiffness and Conditioning of ODEs 252

7.4.2 Conditioning of ODE and DAE systems 254

7.4.3 Modeling Decisions Related to the Index 256

7.4.4 The myth of 'Near Index' Systems 261

7.5 Scaling Variables and Equations 266

7.5.1 Scaling the Variables 267

7.5.2 Scaling the Equations 269

7.6 Automatic Detection of Potential Inaccuracy 276

7.7 Effect of Scaling 278

7.8 Conclusions 280

8 Initial Step Size Selection for Differential-Algebraic Systems 283

8.1 Introduction 283

8.2 Initial Step Size Selection 285

8.3 Scope 288

8.4 Methodology 290

8.5 Consistent initial conditions 292

8.6 Derivatives of algebraic variables 295

8.7 Initial step size 297

8.7.1 Defining the optimal initial step size 299

8.7.2 Initial step size estimator 301
8.7.3 Initial time step combined with step size selection 302
8.8 Implementation within DSL48S 305
8.9 Computational Performance 309
8.10 Conclusions 311

9 Mixed-Integer Dynamic Optimization 315
9.1 Introduction 315
9.2 Problem Scope 317
9.3 Applying MINLP algorithms 319
9.4 Decomposition Approach to MIDO 322
9.5 Casting Batch Process Development as a MIDO 327
9.5.1 Distillation Column Constraints 328
9.5.2 Reaction Constraints 331
9.6 Application of the MIDO Decomposition Algorithm 336
9.7 Summary 338
9.8 Notation 339
9.8.1 Indexed Sets 339
9.8.2 Variables 340
9.8.3 Time Invariant Integer Optimization Parameters 341
9.8.4 Control Variables 341
9.8.5 Time Invariant Continuous Optimization Parameters 341
9.8.6 Parameters 342

10 Conclusions and Recommendations 345
10.1 Screening Models for Batch Process Development 345
10.2 Numerical Issues in the Simulation and Optimization of Hybrid Discrete/Continuous Dynamic Systems 347
10.3 Recommendations for Future Research 349

A Matrix and Vector Norm Proofs 355
A.1 Comments on condition numbers, inf, sup, and rectangular matrices . 357

B Solution of an augmented system of linear equations 359

C Time derivatives of the algebraic sensitivity variables 361

D Review of Batch Plant Design Literature 363
D.1 Multiple Products 366
D.2 Semicontinuous units 367
D.3 Parallel Units 368
D.4 Multipurpose Plants 370
D.5 Varying the Task to Stage Assignment 373
D.6 Discrete Equipment Sizes 374
D.7 Intermediate Storage 375
D.8 Design Under Uncertainty 377

D.9 Solution Methods 379
D.10 Conclusions 382

Bibliography **385**

List of Figures

1-1 Sequential design procedure often used for process development. . . . 30

1-2 Ad hoc iteration iteration strategy employed in an evolutionary approach. 32

1-3 Decomposition algorithm for batch process development. 35

2-1 The two nesting strategies for the performance and structure subproblems investigated by Barrera (1990). 58

2-2 Schematic of the information provided to and produced by the screening formulations. 60

2-3 Plant Superstructure for Batch Reactor 67

2-4 State Task Network for Batch Reaction 67

2-5 The state task network for dynamic optimization of the process development example from chapter 4. This corresponds to the screening model solution obtained from the first process superstructure. 74

3-1 Superstructure for networks of reaction and separation tasks. 80

3-2 Residue curve map for a ternary system with pure components $\boldsymbol{\rho}_1$, $\boldsymbol{\rho}_2$, and $\boldsymbol{\rho}_4$. The fixed point $\boldsymbol{\rho}_3$ represents a maximum boiling binary azeotrope between $\boldsymbol{\rho}_1$ and $\boldsymbol{\rho}_2$. 82

3-3 Ternary system with two distillation regions showing the pot composition trajectory for a feed in distillation region I. 83

3-4 Representation of an arbitrary distillation task by combining sharp distillation cuts and mixers. 87

3-5 Detailed representation of fixed point node e used to derive the purge constraints. 89

4-1 Distillation regions projected onto the facet formed by B, W_1, and P. 118

4-2 Surface defining the upper bound on the extents of reaction given by $f(1-e^{-\kappa t})$. 125

4-3 Process schematic of the solution derived from the superstructure containing only one reaction task. Fixed point flows are given in kmols. . 138

4-4 Process schematic of the solution derived from the superstructure permitting multiple reaction tasks. Fixed point flows are given in kmols. 141

5-1 Process schematic of the solution derived for Case I.A. Streams labels denote the flow of each fixed point in kmols for the campaign. 172

5-2 Process schematic of the solution derived from the superstructure containing only one reaction task. Streams labels denote the flow of each fixed point in kmols for the campaign. 187
5-3 Process schematic of the solution derived from the superstructure requiring all three reaction tasks. Streams labels denote the flow of each fixed point in kmols for the campaign. 190
5-4 Process schematic of the solution derived from the superstructure containing only one reaction task in which the disposal of recycle streams at the end of the campaign is considered. Stream labels indicate the fixed point flows for the campaign given in kmols. 196

6-1 Plot of condenser duty resulting from ABACUSS simulation showing one 'spike' in detail. 205
6-2 Flowchart for the predictor corrector implementation of the BDF method. 211
6-3 Implementation of the dynamic optimization algorithm within ABACUSS. 216

7-1 "Spikes" in the time profile of the condenser duty. 238
7-2 One of the 'spikes' shown in detail. 239
7-3 A comparison of the predicted and corrected solution as a function of the step size during the generation of a spike. 245
7-4 Relationship between the exact Newton update $\Delta\mathbf{x}$, the numerically calculated Newton update $\overline{\Delta\mathbf{x}}$, and the convergence tolerance τ. . . . 249
7-5 Values for x_1 and x_2 for the index-2 system and when $\epsilon = 10^{-3}$. . . . 258
7-6 Demonstration of the difference between $\epsilon = .1$ and the other values of ϵ. 259
7-7 The decay of y onto the high index manifold for different ϵ. 260
7-8 The solution the index-3 system found by solving the equivalent index-1 system (7.40–7.44). 263
7-9 The unstable solution of the index-1 system. 265

9-1 Flowchart of the MIDO decomposition algorithm. 324
9-2 Sequence of subproblem solutions that could be obtained from the MIDO decomposition algorithm. 325
9-3 The superstructure for the MIDO formulation of the process development example from chapter 4. 328
9-4 Decomposition algorithm employed for MINLP problems. 338
9-5 MIDO decomposition algorithm when a convex MINLP screening model is employed. 339

List of Tables

4.1 Constants for the Arrhenius rate expressions for the first order reaction rates ($r_i = C_i k_i e^{\frac{-E_A}{RT}}$). 116

4.2 Azeotrope compositions for the three azeotropes formed between B, W_1, and P. 117

4.3 Product cut sequences for the distillation regions. 117

4.4 Inventory and rental rates for processing equipment. 119

4.5 Material cost, disposal cost, and physical property data for the fixed points. 120

4.6 Raw material costs for the design obtained from the first superstructure.139

4.7 Waste disposal costs for the design obtained from the first superstructure.139

4.8 Utility costs for the design obtained from the first superstructure. . . 140

4.9 Equipment costs for the design obtained from the first superstructure. 140

4.10 Comparison of raw material, waste disposal, utility, and equipment for the design obtained from the first superstructure. 140

4.11 Equipment utilization for the design obtained from the first superstructure. 141

4.12 Raw material costs for the design obtained from the second superstructure. 143

4.13 Waste disposal costs for the design obtained from the second superstructure. 143

4.14 Utility costs for the design obtained from the second superstructure. . 144

4.15 Equipment costs for the design obtained from the second superstructure.144

4.16 Equipment utilization for the design obtained from the second superstructure. 144

4.17 Comparison of raw material, waste disposal, utility, and equipment costs obtained for the second superstructure. 145

4.18 Comparison of the manufacturing costs of the solutions obtained from the two superstructures examined. 145

4.19 Size and approximate solution times for the screening models solved in chapters 4 and 5 on an HP J200 workstation. 147

5.1 Preexponential factors and activation energies defining the rate constants (5.7–5.12) for reactions (5.1–5.6) occurring within the first reaction task 164

5.2 Feasible product sequences for the first case study of the siloxane monomer process. 168
5.3 Composition of the fixed points that are not pure components. 168
5.4 Cost and physical property data for the fixed points. 169
5.5 Inventory and rental rates for processing equipment. 170
5.6 Utility cost data for the siloxane monomer example. 170
5.7 Raw material costs for the entire campaign when minimizing total cost in the first case study. 171
5.8 Waste disposal costs for the entire campaign when minimizing total cost in the first case study. 173
5.9 Utility costs for the entire campaign when minimizing total cost in the first case study. 173
5.10 Equipment costs for the entire campaign when minimizing total cost in the first case study. 173
5.11 Equipment utilization for the design obtained when minimizing total cost in the first case study. 174
5.12 Comparison of raw material, waste disposal, utility, and equipment costs.174
5.13 Feasible product sequences for the second case study of the siloxane monomer process. 177
5.14 Raw material costs for the entire campaign for the process containing only one reaction task. 189
5.15 Utility costs for the distillations for the entire campaign for the process containing only one reaction task. 189
5.16 Equipment costs for the entire campaign for the process containing only one reaction task. 191
5.17 Equipment utilization for the design obtained from the process containing one reaction task. 191
5.18 Comparison of raw material, waste disposal, utility, and equipment costs for the process containing only one reaction task. 191
5.19 Raw material costs for the entire campaign for the process requiring three reaction tasks. 193
5.20 Utility costs for the distillations for the entire campaign for the process requiring three reaction tasks. 193
5.21 Equipment costs for the entire campaign for the process requiring three reaction tasks. 194
5.22 Equipment utilization for the design obtained from the process requiring three reaction tasks. 194
5.23 Comparison of raw material, waste disposal, utility, and equipment costs for the process containing only one reaction task. 194
5.24 Raw material costs for the process considering the disposal of recycled material at the completion of the campaign. 197
5.25 Waste disposal costs for the process considering the disposal of recycled material at the completion of the campaign. 197
5.26 Utility costs for the distillation task in the process considering the disposal of recycled material at the completion of the campaign. . . . 197

5.27 Equipment costs for the process considering the disposal of recycled material at the completion of the campaign. 198
5.28 Equipment utilization for the process considering the disposal of recycled material at the completion of the campaign. 198
5.29 Comparison of raw material, waste disposal, utility, and equipment costs for the process considering the disposal of recycled material at the completion of the campaign. 198

7.1 Value of the local truncation error parameter M in the limits of constant and drastically reduced step sizes. 247
7.2 Numerical statistics for the solution of (7.34–7.36) at different values of ϵ. 261

8.1 Performance of integration code on combined simulation test problems using the initial step length heuristics employed by DASSL. 311
8.2 Performance of integration code on combined simulation test problems using the optimal initial step length calculation. 312

Chapter 1

Introduction

Process modeling technology has changed the way in which continuous/steady state chemical processes are designed and operated (Evans, 1994), yet a similar impact has not yet been witnessed for the design of batch processes. The dynamic nature of batch processing operations coupled with the combinatorial aspects of equipment scheduling and resource allocation dictate that the effective application of process modeling to the design of batch processes is a more formidable task.

Recent advances in modeling capabilities and optimization techniques for dynamic processes now permit the application of detailed modeling technology to batch processes (Barton, 1994). However, the benefits afforded by the application of modeling techniques must outweigh the effort and time required to generate the models, and apply the design methodology. Drawing the analogy to continuous processes, we feel that process modeling techniques can reap the most significant benefits when applied to the design of batch processes by empowering the engineer to exploit interactions between the processing tasks. Modeling enables alternative operating policies to be explored, evaluated, and optimized. However, the systematic design methodologies used for continuous plants do not apply to batch processes, so new methods are required to realize the potential benefits derived from process modeling technology.

This thesis advocates process modeling technology as the basis for an integrated batch process development methodology that can complement and enhance laboratory and pilot scale experimentation. This thesis demonstrates that process modeling technology, employing mathematical models of the physical process at several levels of detail, provides an effective strategy to address the design of batch processes. In particular, the application of process modeling techniques to the optimal development of batch processes has led to the development of *screening models* capable of providing rigorous lower bounds on the cost of the design, and improvements to the numerical integration algorithms employed for solving the simulation experiments. Furthermore, a novel and systematic methodology to address the optimal development of batch processes is presented.

This chapter motivates the development of a systematic methodology employing mathematical models of the processing tasks for batch process design and identifies *batch process development* — the design of a batch process to manufacture a new or modified product in an existing manufacturing facility — as a problem of primary importance. Section 1.1 discusses the economic impact of batch processing, and the importance of batch process development to the specialty chemical and synthetic pharmaceutical industries is covered in section 1.2. Previous approaches that have been applied to the batch process development are then briefly discussed in section 1.3, demonstrating the need for new approaches to the batch process development problem. Although the optimal development of batch process can be expressed as a mixed time invariant integer dynamic optimization problem, no solution techniques to address this class of problems are currently available. This thesis has identified that the key advance that would enable the solution of such problems is the ability to derive models that provide rigorous lower bounds on the design objective. While derivation of such models from the mathematical form of the original dynamic problem formulation may not be possible, alternative models whose solutions provide valid lower bounds for networks of batch reaction and distillation tasks can be derived from engineering insight. These models form the basis for the rigorous decomposition strategy capable of addressing batch process development problem that is introduced in section 1.5. This strategy requires the formulation and solution of two difficult subproblems — a rigorous lower bounding or *screening model* that incorporates the discrete design decisions, and the dynamic optimization of the detailed mathematical models of the process for fixed values of the discrete decisions.

Methods to define and solve these two subproblems are the focus of the two main parts of this thesis. The introduction of the concept of screening models for batch process development is the key idea that enables the mixed-integer dynamic optimization representation of the batch process development problem to be decomposed in a rigorous fashion; the development of screening models is the focus of part 1. In part 2, the numerical integration techniques are improved in order to perform the simulation and optimization of detailed dynamic models more reliably and more efficiently.

1.1 Batch Process Manufacturing

Batch/semicontinuous processes contribute substantially to the global production of chemicals. In fact, Shell (1990) reported that the specialty chemicals and synthetic pharmaceutical industries accounted for \$380 billion of the world's \$1 trillion chemical market in 1988. This contribution is particularly important for developed nations. Developed nations currently enjoy several advantages that favor the production of the specialty chemicals (Polastro and Nystrom, 1993). For instance, the demand for many of these products typically lies within the developed nations, and the impact of labor and energy costs is typically not that high. In addition, for many of these products there are perceived technological barriers which make competition from less developed nations unlikely. This contrasts the commodity chemical market in which the prevailing economic factors favor production in developing nations, particularly those with a cheap energy source. This implies that the importance of batch chemical manufacturing for developed nations is likely to increase as commodity manufacture begins to shift offshore.

Batch processes have achieved a renewed prominence in the chemical process industries due to their suitability for the manufacture of high value added specialty chemicals and synthetic pharmaceuticals. These products are typically required in low volume, and are subject to both short product life cycles and irregular demands. Since such chemicals are often the key active ingredient in many marketed products such as pharmaceuticals, pesticides, dyes, and fragrances, their efficient manufacture is becoming increasingly important to the competitiveness of the chemical process industries (Stinson, 1993).

Batch processes have distinct advantages over continuous processes for the production of low volume products. Since batch processes employ shared, multipurpose equipment, a single multiproduct facility can manufacture many products. Sharing equipment items among products allows for a more efficient deployment of resources and generates cost savings based on economies of scale. In addition, the ability to produce many products in the same equipment provides an operating flexibility not available in continuous manufacturing plants. This flexibility enables the batch plant to respond to fluctuating markets and rapidly advancing technologies, and is largely responsible for its use in the production of specialty chemicals. Production can easily be shifted among products in response to market conditions, and new products may be introduced to existing facilities without significant capital investment.

Batch processing facilities derive much of their flexibility from the strong distinction between the batch *plant* and the batch *process*. The plant refers to the multi-purpose facility itself, while the process refers to the operating procedures and production plans employed to organize the manufacture of different products within the facility. The design of the batch process and the batch plant represent two separate tasks, although the design of one will be strongly influenced by the design of the other.

The design of the plant requires decisions concerning the *superstructure* of the plant. The superstructure is a physical description of the plant equipment, instrumentation, and interconnections. Developing the superstructure requires answering the questions necessary to produce a process and instrumentation diagram. What unit operations should it include? How many of each type of unit should be installed? What size should these be? How should the units be arranged? What interconnecting piping, utilities, and instrumentation should be installed? A typical objective is to answer these questions in a way that maximizes the future flexibility of the plant at minimum cost.

The process design requires the synthesis (or selection) of a sequence of processing tasks to manufacture a product, the definition of operating policies for every task, the allocation and scheduling of plant resources, and the development of detailed operating procedures to implement these tasks in a manufacturing facility. A process must be designed for every product that is manufactured within the plant, yet the design of a process for a particular product may depend on the other products manufactured within the processing facility at the same time.

Most batch plants have a lifetime far greater than the life cycle of the products they manufacture. In fact, the current trend in the specialty chemicals industry is toward the manufacture of products with shorter life cycles and higher functionality that are tailored to specific market niches. Thus, new products are introduced very frequently, and each time a new or modified process design is required. Macchietto (1993) predicts that this trend will accelerate. On the other hand, this trend implies that the expected production requirements of the plant are often unknown at the time of its design, complicating the application of systematic design methodologies for equipment sizing, selection, and plant layout. For these reasons, this thesis has focused on the design of the process, paying particular attention to the batch process development problem defined in the next section.

1.2 Batch Process Development

The goal of batch process development is the design of an efficient process rather than the design of a flexible manufacturing facility. In fact, the new process is usually incorporated into an existing facility. The engineer charged with the development task faces the challenge of designing a large scale process for a recently created or modified product. The information generated from the original synthesis of the product (often an experimental procedure) serves as the starting point. The engineer must derive operating policies for the tasks, and select and schedule the plant's equipment. However, the design of the process is driven by economic factors and constraints not considered at the bench scale. The engineer must also consider issues such as safety, environmental impact, scale effects, and the suitability of construction materials in order to develop a feasible and economic process.

Existing market conditions highlight two motivations for process development to be addressed from a research standpoint. First, these processes must be developed rapidly. In some cases, this provides a competitive advantage by facilitating faster market penetration, by exploiting patent protection to the fullest extent, and by meeting customer expectations. In other cases, such as custom and toll manufacture, rapid process development is required to meet contractual obligations and to compete for new business. Second, these processes must be efficient. Increasing the economic efficiency of manufacture is required to compete on a cost basis; thus, it may increase profit margins or determine if a test marketed product is adopted. Efficient manufacture also permits the revenue stream for a product to continue past the patent expiration, and allows current and expected environmental regulations to be met — both growing concerns in the specialty chemical and pharmaceutical industries (Ahmad, 1997). Moreover, these two objectives, rapid development and high efficiency, are not necessarily mutually exclusive. However, as Laird points out (Stinson, 1993), current development procedures typically only address one or the other. The ultimate objective is to develop efficient batch processes rapidly.

The situation that custom chemical manufacturers often face illustrates the importance of the rapid development of efficient designs. In many cases, a custom manufacturer receives synthesis information for a specific chemical and must define feasible operating policies for the tasks and allocate the resources within their manufacturing facility. Custom manufacturers must be able to solve these problems quickly in order to assess the cost and time required to manufacture the requested product. A manufacturer cannot afford to sign a contract to manufacture a chemical that they cannot produce on their equipment within the allotted time. These producers must comply with contractual obligation to remain in business, so rapid evaluation of the feasibility of the proposed commitments is essential. In addition, they must develop efficient designs to remain competitive.

The urgency for methods and tools specifically aimed at the synthesis and development of batch processes has been recognized in recent years; for example, at Chemical Specialties USA '92 Trevor Laird stated (Stinson, 1993):

> ... custom producers are still under some pressure to control costs as well as to comply with changing environmental and safety regulations. One way in which producers and their clients can meet these needs is by paying closer attention to chemical process development.

Laird also emphasizes the fact that process design is typically subjected to extreme time pressure, so often the most economic or environmentally sound processes are overlooked. The screening models introduced in this thesis employ the available information in a timely manner to identify promising design alternatives at an early stage of the design process. The limited time for development can then be devoted to the most promising alternatives.

1.3 Design Methods for Batch Process Development

The information generated from the original synthesis of a product, often an experimental or pilot plant procedure, serves as the starting point for process development. The synthesis provides the engineer with a sequence of processing tasks capable of transforming raw materials into the desired products along with a feasible sequence of operations that purify the product. In addition, the laboratory scale synthesis provides the engineer with the set of operating policies used for each task at the bench scale. An operating policy is distinguished from a task in the sense that it assigns specific values to quantities, and specific functions to control profiles, rather than a class of similar operations such as "semi-batch operation of Reaction 1." The sequence of processing operations (the *tasks*) combined with operating policies is commonly referred to as the *process recipe*. Most of the previous research in the batch area, typically in the areas of plant design and scheduling, considers the recipe to be fixed a priori, as documented in the review papers of Rippin (1993) and Reklaitis (1989;

1992). Such research aids the engineer facing the process development problem by helping him or her determine a feasible and cost effective allocation of the plant's resources (equipment, labor, and utilities), provided that he or she attempts to implement the recipe developed at the bench scale directly in the manufacturing facility. However, in many cases direct implementation will not be feasible. Moreover, even if it is feasible, direct implementation is typically inadvisable since the objectives of the bench scale experiments differ from those of full-scale manufacture (Allgor *et al.*, 1996). Thus, the engineer may achieve more profitable designs by modifying the recipe during batch process development.

Obviously, the optimal design of a process to manufacture a given product must simultaneously consider changes to the process recipe and to the allocation of facility's resources. Since limited time is available for process development, recipe modifications can only be considered if they are evaluated efficiently. We advocate the use of detailed dynamic models, validated against pilot plant and bench scale experiments, to predict the performance of a particular design. Since the recipe comprises synthesis and design information, the modeling procedure must cope with changes to both.

The synthesis information includes reagent and solvent selection, reaction chemistry, and the structure of the network of processing tasks. Although the reaction pathways and processing steps employed at the bench scale need not remain fixed during the process development, in many cases insufficient information is available to model potential synthesis changes without resorting to detailed bench scale experimentation. Therefore, this thesis does not consider the identification of new solvents and reaction pathways (Knight *et al.*, 1993; Knight and McRae, 1993; Crabtree and El-Halwagi, 1994). However, we consider cases in which decisions involving the selection of reagents and solvents from a list of candidates (see Modi *et al.* (1996) for example) can be systematically evaluated using mathematical models during the process development. In addition, the selection and location of separation stages and the recycle structure are considered during the process development. The synthesis decisions typically involve selecting from a set of discrete choices, where different dynamic models may be employed to describe each.

The process design specifies the operating policies for the processing tasks defined at the synthesis stage and a feasible allocation of the manufacturing facility's resources. For a given equipment assignment, the effect of changes to the operating policies of the tasks can be predicted using dynamic simulation or dynamic optimization. In the remainder of this section, we consider the general approaches that have been applied to the process development problem, and demonstrate that the problem in which we are interested can be formulated as a mixed-integer dynamic optimization problem.

Batch processes have typically been designed using a sequential procedure, similar to the one shown in figure 1-1, that begins with the discovery of a new product in the laboratory. The engineer charged with the development task then determines feasible operating policies for the tasks in the manufacturing-scale equipment and a feasible allocation of the manufacturing facility's resources for production. Although the decisions made at all three stages of the design effect the efficiency of the process, most batch process design research has considered the process recipe to be fixed (Rippin, 1993), focusing on the third stage of the sequential design procedure. Only a few researchers have examined methods to incorporate recipe modifications during the design of a batch process (discussed in chapter 2), and to date, none have proposed rigorous techniques that can handle the discrete and dynamic operating decisions simultaneously.

In many situations, the partitioning between the process synthesis and the latter stages of process development arises naturally. This is commonly the case with custom chemical manufacturers who are contracted to deliver a specific chemical to fulfill an order from a single customer. In many cases, the custom manufacturer receives the synthesis information for the product and is left with the task of defining feasible operating policies and allocating the resources within the manufacturing facility. In large chemical companies, organizational boundaries may implicitly require the separation between the synthesis and design stages of the process development. For instance, many companies have separate departments, sometimes located at different sites, dedicated to research and process engineering. This separation restricts the in-

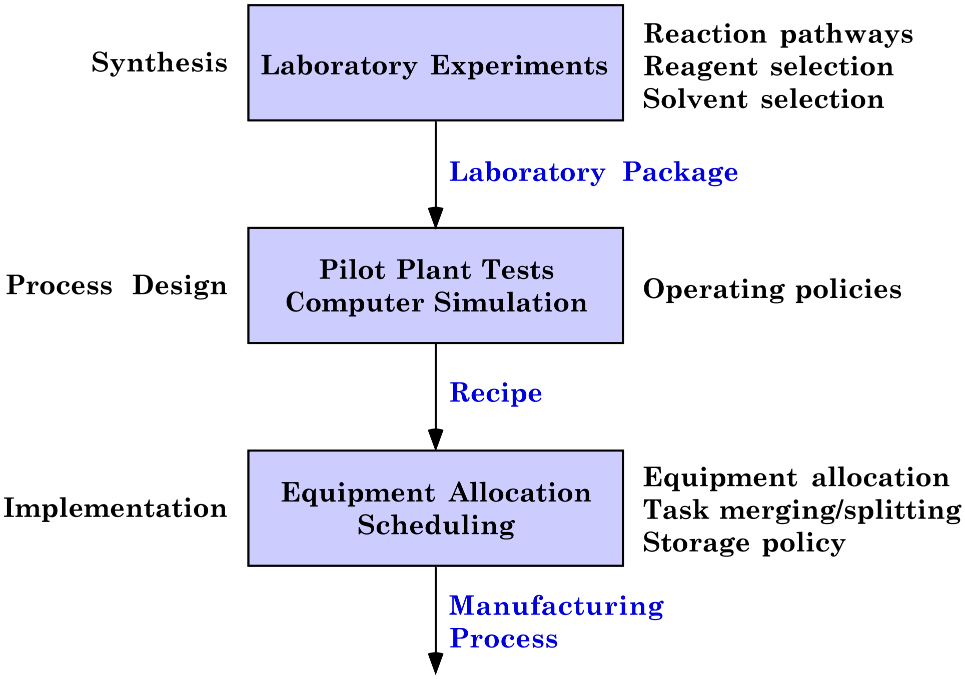

Figure 1-1: Sequential design procedure often used for process development.

tegration of design tasks; more complete integration of the design process requires a change in the structure of manufacturing organizations (Reklaitis and Preston, 1989). Until such changes are realized, many processes will be designed while the process synthesis information remains fixed. However, even if the synthesis is separated from the rest of the design, the development of the operating policies and equipment allocation should not be partitioned.

Barrera and Evans (1989; 1990) demonstrated that the ability to modify the process recipe, both to improve performance and ensure feasibility of the processing tasks, is critical to the success of the design. They decomposed the process development problem (without the synthesis aspects) into the *performance* and *structure* subproblems based on the nature of the decisions addressed in each subproblem; these subproblems are analogous to the final two tasks in figure 1-1. The objective of the performance subproblem is to determine optimal operating policies for the sequence of processing tasks once the plant resources (e.g., equipment, labor, and utilities) have been assigned. The structure subproblem seeks to find the optimal allocation of plant resources after the process recipe has been fixed, and involves both continuous and discrete decision variables, but contains no process dynamics. Methods are currently available for the solution of each of these subproblems. On the one hand, the performance subproblem defines a dynamic optimization problem. Solution of this subproblem requires detailed dynamic models of the processing tasks, or the ability to evaluate the operating policies using extensive experimentation. Charalambides *et al.* (1993) demonstrated that the performance subproblem can be represented and solved as a multistage dynamic optimization problem, once the processing structure and control variables have been selected. They have applied this technique to several examples (Charalambides *et al.*, 1995a; Charalambides *et al.*, 1995b; Charalambides, 1996). On the other hand, the structure subproblem represents a combinatorial optimization problem that can be addressed using mixed-integer linear or nonlinear programming techniques. Since the process will typically be operated in campaign mode, the structure subproblem represents a problem that has been addressed by both the batch scheduling and batch plant design literature (Reklaitis, 1989; Reklaitis, 1992; Rippin, 1993).

Although established techniques now exist to solve both subproblems in isolation, to date no methods exist to address them simultaneously. At best, ad hoc iterations between the two subproblems have been performed, resulting in an evolutionary procedure for the improvement of a 'base case' design (Barrera, 1990; Salomone *et al.*, 1994). Barrera's approach iterates between the performance and structure subproblems, fixing the variables used in one subproblem while the other subproblem is solved; i.e., the performance is optimized for a given structure, and the structure is optimized for fixed operating policies, as shown in figure 1-2. He demonstrated the significant benefits that could be gained by considering the optimization of both resource allocation and operating policies together, even using an ad hoc procedure.

With this iteration strategy, either subproblem can be solved to optimality every time the variables in the other are updated, placing one subproblem in an outer optimization loop and the other in an inner loop. Placing the performance subproblem in the outer iteration loop yields a local improvement strategy for the initial design; iterations are terminated based on the lack of improvement in the current solution.

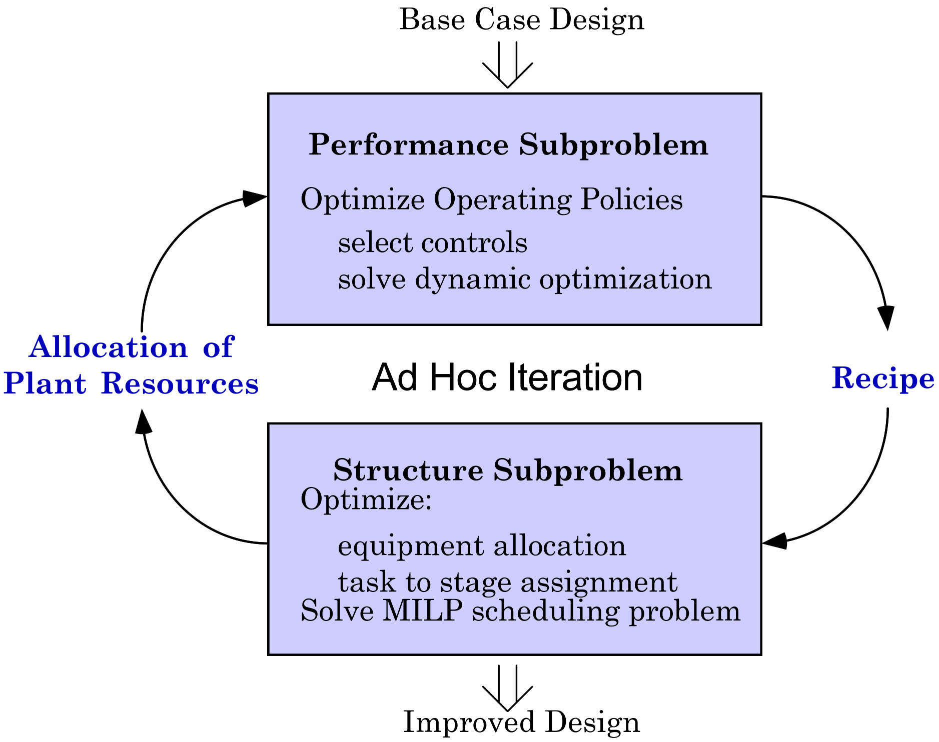

Figure 1-2: Ad hoc iteration iteration strategy employed in an evolutionary approach.

At termination the original design has been improved, but no information is available to indicate how close this design may be to the global optimum or to suggest whether further optimization is warranted. Placing the structure subproblem in the outer iteration loop permits enumeration of the discrete space, but provides no way to prune the discrete space, making total enumeration inevitable.

In order to avoid total enumeration of the discrete space, rigorous lower bounds on the cost of the overall design are required. Although the structure subproblem is incapable of providing such bounds, this thesis employs engineering insight to derive lower bounds on the production cost for networks comprised of batch reaction and distillation tasks. These models are introduced in the next section. They permit the derivation of a rigorous iteration strategy for the improvement of batch processes that is introduced in section 1.5.

1.4 Screening Models for Batch Process Development

This thesis introduces the concept of *screening models* for batch process development. Screening models yield a rigorous lower bound on the cost of production, providing both design targets and a valid way in which to prune or screen discrete alternatives (process structures and equipment configurations) that cannot possibly lead to the optimal solution. These models consider changes to the process structure, the operation of the tasks, and the allocation of equipment simultaneously. In addition, these models embed aspects of the process synthesis not considered in previous research dealing with batch process design. However, they do not provide a detailed process design, so they must be used in conjunction with techniques that consider the dynamics of the process in detail, such as the multi-stage dynamic optimization formulations used to address the performance subproblem (Charalambides, 1996).

Screening models provide targets for the design of batch processes which can either be used in isolation, used to enhance existing approaches, or used as the foundation for a rigorous decomposition strategy for the solution batch process development problems. In isolation, the solution of the screening model may be all that is needed to determine whether it is worth pursuing further development of a new product. If the product is not profitable given a lower bound on the manufacturing costs, then there is no need to pursue further design or experimentation. Screening models provide a design target to which the solutions from the sequential or evolutionary approaches may be compared. This comparison can be used to assess the potential benefits of continued optimization. Since the evolutionary approach is merely a local search technique, the solution of the screening model may indicate whether the iteration should be attempted from another initial point. If another sequence of iterations is justified, the solution provides a prime candidate for the initial point of this sequence.

Screening models can also be used to identify a set of candidate solutions which may have a lower cost than a given base case design. The performance problem can then be solved for each of these discrete alternatives. Used in this fashion, the screening models provide a rigorous way to prune the space of discrete alternatives. In addition, the solution of the screening model provides good initial guesses and a feasible processing structure for the multistage dynamic optimization problem solved to obtain a detailed design. This point is discussed in more detail in section 2.4. Although the screening models can be employed merely to identify candidates for enumeration, their lower bounding properties can also be exploited to derive a rigorous decomposition algorithm to address batch process development.

1.5 Rigorous Decomposition Algorithm

Screening models also enable the derivation of a rigorous decomposition strategy for batch process development that is detailed in section 2.4. The strategy is quite simple and is diagrammed in figure 1-3. First, the screening model is solved to provide a lower bound on the solution of the corresponding performance subproblem (this is a lower bound on the global solution on the first iteration). The solution of the screening model also provides values of the binary variables satisfying all of the logical constraints (e.g., equipment allocated to performed tasks, equipment assigned from available inventory, etc.) and initial guesses for the material flows and control profiles for the dynamic optimization. The performance subproblem, a multistage dynamic optimization, is then solved. The solution of this problem represents a feasible design, so if it is better than all of the designs that have been found so far, we update the value of the objective. We add an integer cut to the screening model to exclude the solution just found and solve the screening model again. After each solution of the screening model, we check to see if either the problem is infeasible or the solution is greater than the best solution of the performance subproblem found so far. If either of these is true, we terminate the iteration with the confidence that we have rigorously searched the space of discrete alternatives.

Since this thesis considers campaign manufacture in which every equipment item is dedicated to a specific task (or set of sequential tasks) and allocated for the duration of the campaign, the equipment assignment remains fixed over the entire campaign.

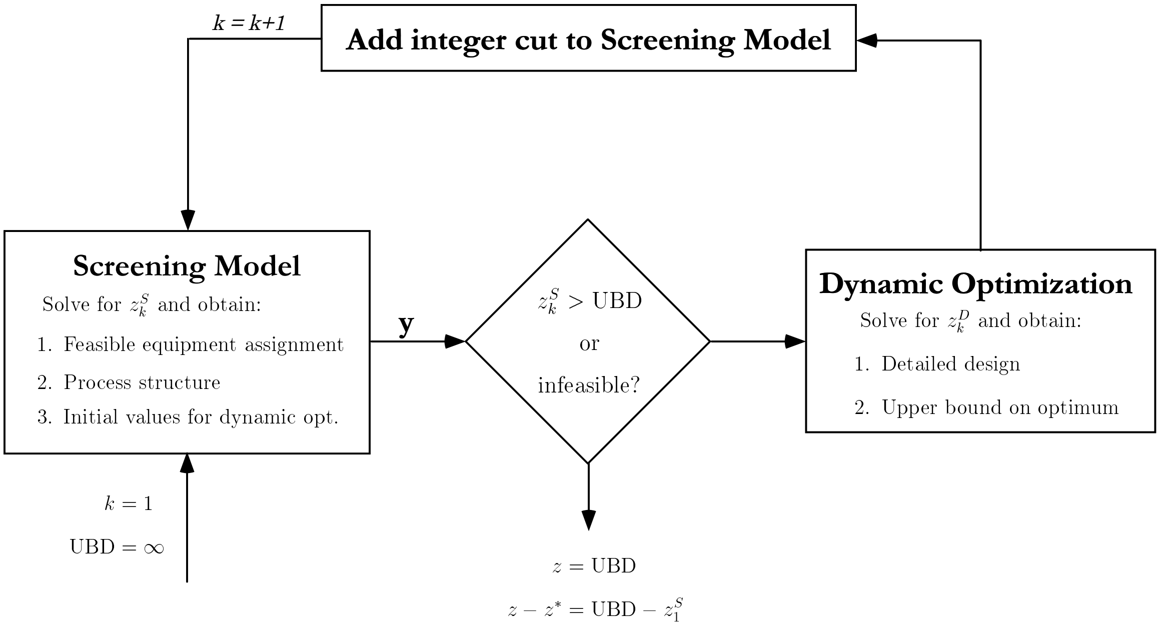
k = k+1
Add integer cut to Screening Model
Screening Model
Solve for z_k^S and obtain:
1. Feasible equipment assignment
2. Process structure
3. Initial values for dynamic opt.
y
$z_k^S > \text{UBD}$
or
infeasible?
Dynamic Optimization
Solve for z_k^D and obtain:
1. Detailed design
2. Upper bound on optimum
$k = 1$
$\text{UBD} = \infty$
$z = \text{UBD}$
$z - z^* = \text{UBD} - z_1^S$

Figure 1-3: Decomposition algorithm for batch process development.

In addition, every batch is processed in exactly the same fashion and end effects are ignored during the optimization of the process. These assumptions imply that the integer decision variables are fixed for the duration of the entire campaign, so they can be represented as time invariant parameters that are restricted to $\{0, 1\}$ within the dynamic optimization. Thus, the dynamic optimization problem representing the performance subproblem can be augmented with the constraints of the structure subproblem to yield a mixed time-invariant integer dynamic optimization (MIDO) problem (Allgor and Barton, 1997b); MIDO problems are discussed in detail in chapter 9, and the batch process development example from chapter 4 is formulated as a mixed time invariant integer dynamic optimization problem to demonstrate this point.

As discussed in chapter 9, the reason that well known decomposition approaches used for mixed-integer nonlinear programming (MINLP) problems cannot be extended to the MIDO problem is that valid constraints for the Master problem cannot be derived from the mathematical form of the primal problem (the dynamic optimization). Therefore, the key to deriving a rigorous decomposition strategy for the MIDO problem is the ability to formulate a model that defines rigorous (and useful) lower bounds on the objective function, that overestimates the space of feasible designs, and that can be solved to guaranteed global optimality. However, we have already mentioned that the screening models provide valid lower bounds for the solution of the MIDO representation of the batch process development problem. Thus, the same decomposition strategy can be applied to other classes of mixed time invariant integer dynamic optimization problems, provided that suitable screening models can be derived.

The decomposition algorithm requires models at two very different levels of detail. The screening models are algebraic models that contain limits of the performance of the dynamic process and address the discrete design decisions. On the other hand, the detailed dynamic models of the processing tasks employed within the performance subproblem represent the processing tradeoffs as accurately as possible. As might be expected, the tools and expertise needed to address each of these problems also differs. The subproblems within this algorithm motivate the parts of this thesis.

Engineering insight and combinatorial optimization are required for the formulation and solution of the screening models. The formulation and solution of these models is the focus of the chapters contained in the first part of this thesis. On the other hand, the solution of the performance subproblem requires robust techniques for the solution of hybrid discrete/continuous differential-algebraic systems. The advent of sophisticated equation based modeling environments (Barton, 1992) coupled with the increasing availability of libraries of dynamic models facilitate the definition of the performance subproblem, but the requirement that these models must be solved accurately, efficiently, and robustly places severe expectations on the numerical integration techniques. The application of state-of-the-art hybrid discrete/continuous simulation languages to the simulation and optimization of batch reaction and distillation tasks has uncovered several previously unreported numerical problems encountered during solution of the initial value problems (IVP) required for both dynamic simulation and optimization. Part 2 of this thesis identifies and mitigates some of these numerical problems, improving both the robustness and efficiency of the numerical integration code. These improvements become particularly important when solving dynamic optimization problems, since the integration code must be robust enough to deal with the automated manipulation of control profiles without user intervention.

1.6 Numerical Issues in the Detailed Simulation of Batch Processes

As has been recognized for some time (Fruit *et al.*, 1974), batch processes are characterized by both discrete and continuous dynamic behavior. While phenomena such as the mass, momentum, and energy balances can be described by continuous dynamic models, the control actions required to drive these models through the scheduled operation of the processing tasks impose a set of discrete changes. Discrete changes also arise naturally due to physical changes such as the appearance and disappearance of phases. Thus, combined discrete/continuous dynamic models are required to represent the detailed behavior of batch processes. Any suitable simulation environment must provide facilities to represent both aspects of the behavior and provide robust techniques for the solution of the resulting models.

The development of simulation methods to address batch processes has evolved along similar lines to general techniques for combined discrete/continuous simulation. The initial tools developed for the simulation of batch processes (Fruit *et al.*, 1974; Joglekar and Reklaitis, 1984; Czulek, 1988) augmented discrete event simulators (Pritsker and Hurst, 1973; Pritsker, 1986; Sim, 1975) with limited continuous dynamic modeling capabilities, usually in the form of models for specific processing steps. On the other hand, more recent developments have added discrete event capabilities to sophisticated continuous dynamic modeling languages such as Speedup (AspenTech, 1993) and DYNSIM (Sørensen *et al.*, 1991). Barton (1994) provides a review of these technologies. While the former class has proven to be a useful complement for production planning and scheduling tools that employ more abstract models, extension to process development problems has proven problematic, even by people who have touted the benefits of such tools (Terry *et al.*, 1989).

For several reasons we feel that the detailed modeling and optimization of batch processes required for batch process development necessitates the use of sophisticated dynamic modeling environments augmented with discrete capabilities (e.g., ABACUSS (Barton, 1992)). The modeling environment decouples the description of the model describing the behavior of the physico-chemical transitions occurring within the equipment units from the sequence of control actions imposed on the process. Regardless of the nominal mode of operation, only one model of the physical description of the system needs to be developed. Processing operations are described by deriving schedules comprised of task entities to represent the external actions applied to the system. This decomposition into the model of physical behavior and the schedule of external actions allows a given physical model to be reused under many different operating scenarios. The discrete attributes are represented by changes to the functional form of the system of differential-algebraic equations describing the continuous dynamic behavior. This decomposition facilitates the modeling of semi-batch, semi-continuous, and continuous units along with those operating in a batch mode within a single environment. It also permits the modeling of processes in which the integrity of batches is not maintained.

These environments permit individual tasks to be simulated in isolation, but more importantly, they permit detailed analysis of the dynamic interactions between processing tasks, as demonstrated by several examples reported in the literature (von Watzdorf *et al.*, 1994; Winkel *et al.*, 1995). In particular, modeling the entire batch process permits a systems approach to the process design. System simulation is required to assess the process alternatives considered during the design of integrated batch processes, especially those processes containing recycles of material from one batch to another. For example, batch processes designed for pollution prevention may recycle cuts from a batch column to an upstream reaction task (Ahmad and Barton, 1994). System simulation is also required to optimize integrated processes in which processing tradeoffs between upstream and downstream tasks are exploited, such as those considered by Charalambides (1996) and those considered within this thesis. The dynamic interactions between processing steps can be as simple as a model of a reaction vessel with an overhead condenser, yet they may complex enough to consider an entire batch process in which not only the main processing steps are considered, but also the detailed dynamic interactions between different equipment units, the interconnecting piping, valves, and pumps are modeled. Thus, the environment permits a convenient framework in which to model and evaluate the operating procedures that will be carried out by the plant operators and control system.

Combined discrete/continuous modeling environments also provide the flexibility required to model the batch process at an appropriate level of detail. Models are constructed from the equations representing the physical behavior of interest. Simple models are then combined in a hierarchical fashion to construct models of more complex phenomena. As demonstrated by Allgor *et al.* (1996), this modeling flexibility is required for the scale-up of a batch process from the laboratory to manufacturing equipment. For example, the heat transfer equipment and geometry of the manufacturing vessels may dictate the feasibility of proposed operating policies. Thus, a basic model of the processing behavior must be easily adapted to suit the performance of tasks in specific items of equipment and to model tasks that may not be available from a standard library of operations. For example, batch distillation simulations can be posed using models of varying complexity that can be tailored to represent the specific type of heat transfer equipment, control system, and column configuration (e.g., rectifier, stripper, or middle vessel (Davidyan *et al.*, 1994)) that exist in the actual manufacturing facility. ABACUSS simplifies the maintenance of models at different levels of detail through the use of model inheritance, permitting a basic model of the physical behavior to be refined to suit a particular item of equipment (Barton, 1992). Modeling flexibility is also required for a quite different reason during the development of batch processes. In many cases, a limited amount of information is available at the start of the development process. Thus, the models of both the physical properties and the behavior of the system may be quite simple at the start of the development process. For instance, only mass balances and crude approximations of the processing times of the tasks may be required at the initial stage of the design. As more information becomes available, more detailed models may be employed. Thus, each of the basic processing steps in a manufacturing process may be represented by a set of models, varying in the level of detail, before the design is completed. Furthermore, it may not be possible or cost effective to obtain data (VLE, reaction kinetics, etc.) that may be required for the application of the most detailed models. Therefore, the modeling environment must provide the flexibility to combine detailed and simple models not only during different stages of the model development, but also during a particular simulation experiment.

Combined discrete/continuous modeling environments such as ABACUSS meet all the requirements outlined above, and we believe that they are the only technology available that is suited to address the detailed modeling of general batch processes. Furthermore, the equation-based representation of the models is well-suited to the application of dynamic optimization techniques. These environments incorporate useful features from the standpoint of model development and flexibility, but they require knowledgeable users to take full advantage of their capabilities because proper model construction and specification of a correct simulation experiment are both nontrivial tasks. Not only are features to analyze the index of the DAEs (Feehery and Barton, 1995) and to assist with the specification of initial conditions (AspenTech, 1993) required, but also facilities to analyze the structure and degrees of freedom during the model development would be useful. However, the demands placed on the users of such systems pales in comparison to the expectations placed on the numerical codes employed to solve these generic combined discrete/continuous problems. The modeling environments draw their flexibility from the separation between the description of the model and the numerical techniques employed to solve the simulation experiments, which is precisely what places severe demands on the numerical solution techniques. The numerical analysis portion of this thesis has grown out of the need to improve the accuracy, efficiency, and robustness of the numerical procedures used to solve the discrete/continuous dynamic models of the batch processing tasks required for the design of detailed operating policies.

Using ABACUSS to simulate the batch distillation of wide-boiling azeotropic mixtures has uncovered some previously unreported numerical difficulties that are described in chapter 7. We have determined that the problems observed indicate a breakdown in the integrator's error control strategy, demonstrating that the potential exists for inaccurate results to be obtained without any warnings issued by the integration code. This research identified the source of the numerical difficulties as an ill-conditioned corrector matrix. We have developed a strategy to guarantee the accuracy of the solution to the mathematical model in spite of the fact that the computations are performed on machines of finite precision. Chapter 7 derives a strategy that automatically determines the optimal scaling the variables and equations of the models during the integration. This reduces the effect of ill-conditioned models and provides the modeler the freedom to work with a convenient set of units when writing the models. When used in conjunction with automatic differentiation techniques, it permits the automatic determination of the effects of the rounding error on the solution of the corrector iteration. This allows the integration code to automatically detect simulations in which the potential exists for the integrator's error control procedure to break down.

Given the limited time available for process development, efficient solution techniques are required for integration and dynamic optimization of detailed process models. Therefore, we have improved the efficiency of the numerical integration techniques available for the type of models in which we are interested. The well known differential algebraic equation code DASSL (Petzold, 1982a) was tailored for large sparse unstructured systems as part of this research. The resulting code has been called DSL48S (Feehery *et al.*, 1997). The code employs the MA48 linear algebra routines, works with a combined analytical and numerical Jacobian matrix, and has incorporated the automated scaling algorithm in chapter 7. The code also contains an efficient method for sensitivity analysis that was developed by Feehery and Barton (1997). In addition, the code employs a new method to start the backward differentiation formula integration codes efficiently, an important feature when solving discrete/continuous systems. This method is described in chapter 8. The method consists of two main steps. First, the time derivatives of the algebraic variables and the second order time derivatives of the differential variables are determined at the initial time. We define criteria for the optimal initial step size, and demonstrate that the information provided by the second order time derivatives of the differential variables can be used to estimate this optimal initial step length. The second step of the procedure simultaneously determines the optimal initial step length and the values of the system variables at this step length by augmenting the system of equations solved during the corrector iteration. This method improves the efficiency of the integration code during the initial phase of the integration and substantially reduces the number of convergence and truncation error failures encountered.

1.7 Outline of Thesis

The thesis is divided into two parts. Each part focuses on techniques for the formulation and solution of one of the two subproblems involved in the decomposition approach introduced above. The first part emphasizes the formulation and application of the screening models to batch process development. The second part focuses on improvements to the numerical solution techniques employed for the integration of the discrete/continuous dynamic models.

The first part of this thesis focuses on the derivation and application of screening models for batch process development. Chapter 2 reviews the previous research that has addressed batch process development and motivates the development of screening models. Section 2.4 describes the decomposition algorithm for batch process development in more detail. Chapter 3 develops screening models for networks of batch reaction and distillation tasks. We prove the bounding properties of the models for the types of processes considered. We show that these models can be cast as mixed-integer linear programming problems. Chapters 4 and 5 demonstrate the application of the screening models to case studies. The case studies also show how reaction targets can be derived and incorporated into the models.

The second part of this thesis improves the numerical solution procedures for the hybrid discrete/continuous initial value problems. Chapter 6 illustrates the numerical difficulties that motivated this portion of the research and reviews some of the mathematical background required to understand the subsequent chapters. Chapter 7 proves that the observed numerical difficulties are caused by an ill-conditioned iteration matrix, and explains how the integration codes error control strategy can permit the generation of 'spikes.' Chapter 7 also derives an automated technique to scale the iteration matrix, mitigating the effects of ill-conditioning, and proves that this scaling comes very close to the optimal scaling for the sparse unstructured matrices with which we are concerned. Chapter 8 derives a novel and efficient method for starting the DAE integration codes employed for the solution of the IVPs encountered during hybrid discrete/continuous simulation and optimization. Chapter 9 defines mixed time invariant integer dynamic optimization problems and illustrates that conventional MINLP algorithms cannot be extended to this class of problems. However, the decomposition strategy for batch process development can be extended to this class of problems provided that suitable screening models can be derived. We prove the correctness of the decomposition algorithm, and illustrate that batch process development can be cast as mixed time invariant integer dynamic optimization problem.

Chapter 2

Batch Process Development

Batch process development is encountered frequently in the specialty chemical and synthetic pharmaceutical industries. Process development requires the design of a manufacturing process for a new or modified product in an existing manufacturing facility. The engineer's ability to design an efficient batch process that fits into the available equipment rapidly is critical to the success of many specialty chemical manufacturers (Allgor *et al.*, 1996).

Traditionally, changes to the process recipe have not been considered, and a sequential design procedure has been employed (see figure 1-1). The process synthesis and operating decisions are made at the bench and/or pilot plant scale, and then the engineer allocates and schedules the equipment in the manufacturing facility for production. Recently, researchers have considered employing mathematical models of the processing tasks to evaluate the impact of recipe modifications during process development. Their research, reviewed in the next section, highlights the benefits provided by performing recipe modifications in conjunction with the allocation of plant resources. However, none of this research considers rigorous methods for the simultaneous optimization of the discrete and continuous decisions encountered during batch process development. This thesis addresses this deficiency.

The screening formulations derived in this work address the discrete and continuous decisions encountered during process development simultaneously. The proposed screening formulations provide bounds on the best attainable process design by optimizing the process recipe and equipment allocation concurrently. The resulting models optimize the processing structure and the allocation of plant resources in detail by replacing the detailed dynamic performance models with targeting models guaranteed to provide lower bounds on the design cost and to overestimate the feasible region of operation. Furthermore, these models can be solved with reasonable computational effort to guaranteed global optimality. The screening formulations are incorporated within a design methodology that permits detailed treatment of the continuous operating decisions as well, allowing an engineer to perform *optimal* batch process development. The approach introduces a novel way in which performance bounds based on engineering insight can be combined with detailed discrete/continuous models of process dynamics and sophisticated dynamic optimization algorithms to yield a systematic methodology for batch process development. The procedure considers both the discrete and continuous design decisions and incorporates some elements of the process synthesis during the process design. Chapter 3 describes how the desired bounding property is preserved during the formulation of the screening models. The rigorous lower bounds provided by these models also enables a rigorous decomposition algorithm for optimal batch process development to be derived. This algorithm, which is discussed in section 2.4, represents the first rigorous and systematic methodology for the optimization of these processes.

2.1 Previous Research

Allgor *et al.* (1996) clearly demonstrated the industrial importance of batch process development, and stressed the need to develop methodologies to address process development in a systematic fashion. The ability to modify the process recipe in order to improve the performance or ensure the feasibility of the processing tasks is critical to the success of the design obtained. In fact, Rippin (1993) highlighted both the importance and difficulty of varying task performance during batch process design, and chronicled the lack of attention that the problem has received. Despite its importance, only a few researchers have examined systematic methods to incorporate recipe modifications during the design of a batch process, and to date, no one has proposed techniques to consider the discrete and dynamic operating decisions simultaneously. We will examine the existing research in two categories. First we briefly examine the research that considers the recipe fixed a priori, and highlight what elements of this research can be applied to the development problem. Next, we assess the applicability of the research that has considered recipe modifications to the process development problem.

Partitioning the research into these two categories follows naturally from the sequential approach often used by industrial manufacturing concerns to develop a new process. The typical sequence of events is shown in figure 1-1. First, a new or modified product is discovered in the laboratory. Next, improvements in the chemical synthesis and product purification are performed at the bench scale before presenting the engineers with a process recipe. The engineers may then decide to test the operating policies they receive for feasibility and make minor modifications based on experience or other analysis tools; for instance, suitable reflux ratios for the columns could be determined using Batchfrac (AspenTech, 1991). Once the operating policies are satisfactory, the final process recipe is implemented in the production facility on the available equipment in the most cost effective manner. Existing research focuses on one of the steps in this sequential procedure.

As previously mentioned, partitioning the design decisions into two sets, those defining the operating policies of the tasks and those defining the allocation of the facility's resources, decomposes process development into the performance and structure subproblems. The existing research can only address variations on either one of these two subproblems. At best, ad hoc iterations (shown in figure 1-2) between the two subproblems have been performed. The previous research in this field will be discussed according to its relation with the structure and performance subproblems, and strategies designed to couple the decisions in the two subproblems will also be covered.

2.1.1 Design with Fixed Recipes

Most of the research related to batch process design considers the recipe to be fixed. Thus, aspects of this research may apply to the structure subproblem encountered during process development. This research can be broadly classified into the batch *scheduling* and *plant design* problems, and some of the techniques used for each problem can benefit process development.

In the typical batch plant design formulation, the installation cost of plant resources is minimized subject to a fixed set of production requirements and fixed recipes. This deterministic design problem was first addressed by Ketner (1960) and later by Loonkar and Robinson (1970; 1972). The original formulations of the batch plant design problem considered only simple operating scenarios. Subsequent research has considered more complicated scheduling aspects, design of multiproduct and multipurpose plants, uncertainty in the production demands and process performance, and the selection of equipment in discrete, rather than continuous, sizes (Rippin, 1993). The progress on this problem has been reviewed by Reklaitis (1989) and Rippin (1993). The growth in the list of publications since Rippin's previous review (1983a) demonstrates that a significant amount of research has been conducted over the last ten years. However, progress in these areas has been incremental, and to this date a rigorous formulation of the problem that accounts for all possible alternatives has not been found; appendix D reviews this literature in more detail. Hence, in his most recent review, Rippin (1993) characterized the progress in this research as "filling in the holes." In addition, little effort has been devoted to questioning the fundamental assumptions of the plant design problem. This is disappointing since many of these assumptions severely limit the application of the technology. For instance, only limited uncertainty in the production demands placed on the plant over its lifetime have been considered, yet the life cycle of the products is usually far shorter than the lifetime of the plant. Many products are subject to quickly changing markets or may be displaced by rapidly advancing technologies, so the products that will be manufactured in the plant near the end of its lifetime are most probably unknown at the time of its design. This fact has not been addressed in the literature, although organizations considering investment in a new multipurpose facility are forced to confront this problem.

The batch plant design problem typically assumes that the products will be manufactured in campaigns, either in single product campaigns, or mixed product campaigns (Birewar and Grossmann, 1989) with either single or multiple production routes (Faqir and Karimi, 1989; Faqir and Karimi, 1990). Since the products considered in the batch process development problem will also be manufactured in campaigns, the scheduling and equipment allocation techniques created for batch plant design can be applied to process development. In addition, the equipment allocation and scheduling constraints developed for the plant design problem can handle some of the complications that arise from implementing the process in an existing manufacturing facility. In particular, Knopf *et al.* (1982) introduced processing times that depend on both the equipment item and the batch size, a necessity when dealing with the recipe modifications considered in process development. In addition, the use of an existing facility dictates that the equipment must be chosen from an inventory of available items. The allocation constraints in this situation are similar to those developed to address plant design when the equipment items are only available in discrete sizes (Voudouris and Grossmann, 1992a).

Although the objectives of the plant design and the batch process development problems are different, the constraints related to the allocation of equipment are very similar because both problems address campaign manufacture. In many cases, the plant design problem contains both discrete and continuous variables, but contains no dynamic behavior. This permits the use of MILP and MINLP optimization procedures to solve the resulting plant design formulations. Heuristic, mathematical, and hybrid optimization techniques have been applied to the solution of these formulations. In most cases, the ability to solve the resulting optimization problem, rather than the ability to pose the constraints, governs the complexity of the design possibilities considered.

The batch process scheduling problem has also received a lot of attention in the academic literature (Reklaitis, 1989; Reklaitis, 1991; Reklaitis, 1992; Pekny and Zentner, 1993; Pantelides, 1993). Given a fixed set of demands and fixed process recipes, the available plant resources are allocated in an optimal fashion over a given time horizon. Initial approaches for the scheduling problem considered either flexible operating scenarios using heuristic or approximate methods to optimize the operation or found exact solutions under more restrictive operating configurations. The two major challenges in the short term scheduling of batch plants is finding a mathematical representation that permits fully general operating configurations, and finding efficient solution techniques to solve the models. The former can be met by abstracting the batch process as a state task network (Kondili *et al.*, 1988; Kondili *et al.*, 1993) or resourced task network (Pantelides, 1993), uniformly discretizing the time domain, and casting the problem as a mixed integer linear program using general discrete time scheduling techniques (Papadimitriou and Steiglitz, 1982). The disadvantage with these formulations is that the time discretization must be established prior to the solution procedure so that all processing events start and end on a boundary between time intervals. This results in formulations with many discrete variables that are difficult to solve. Advances in the solution methods for these problems have led to tailored branch and bound procedures and tighter problem formulations that enable some reasonably sized problems to be solved (Shah, 1992; Shah *et al.*, 1993). Continuous time scheduling formulations, commonly employed in the operations research community (Blazewics *et al.*, 1991), can reduce the number of discrete variables required in batch scheduling problems (Xueya and Sargent, 1994; Pinto and Grossmann, 1995; Schilling and Pantelides, 1996), but they are not yet as robust as the discrete time algorithms and still require partitioning of the time horizon into a number of intervals that exceeds the number of events that occur over this time at the optimal solution.

The flexible operating configurations afforded by the discrete time scheduling formulations are more than is necessary for the processes considered in the batch process development problem. Process development assumes that the products will be manufactured in campaigns, and every batch will follow the same path through the processing equipment. Provided that batch size dependent processing times are taken into account, short term scheduling techniques can be applied to the development problem, but the difficulty in solving the resulting models is probably not warranted because the modeling flexibility is not needed. However, the state task network representation of the process developed for the short term scheduling problems does provide a convenient framework for defining the multistage optimal control problems that can be used to optimize the operating policies for a given equipment configuration.

2.1.2 Design with Recipe Modifications

The objective of recipe modifications is to increase the process efficiency by exploiting tradeoffs between the operating cost and the time profiles of key operating variables and the values of key operating parameters. Recipe modifications have been considered as part of the plant design and process development problems. In both cases, existing research addresses slight variations on the performance subproblem proposed by Barrera (1990). The performance subproblem determines the optimal operating policies for the processing tasks given a fixed allocation of plant resources and a set of design constraints (product purity, limiting temperatures, pressures, etc.). For example, a typical instance of the performance subproblem could be stated as follows:

> A process consisting of a single reaction and distillation task has been synthesized for the manufacture of a particular product; mathematical models are available to predict the performance of the operating policies. A 500 gallon stainless steel reactor has been dedicated to the reaction task, and a 500 gallon packed distillation column with eight theoretical stages has been assigned to the distillation task. Determine the reagent and solvent feed policies for the reaction task, reflux policy for each distillation cut, the time-averaged flows for any recycled material, and the location of all the product and off cuts that minimize the per unit production cost of the desired product.

The performance subproblem requires dynamic models of the processing tasks (assumed in the example above), or the ability to evaluate the operating policies using extensive experimentation. Further, it can be solved as a multistage dynamic optimization problem provided that models are available and the control variables have been selected (Charalambides *et al.*, 1995a). For the results to be meaningful the models must accurately represent the complicated dynamic behavior of the processing tasks.

A large volume of research has addressed the optimization of isolated processing tasks, particularly batch reactors and batch distillation columns (Rippin, 1983b; Hatipoglu and Rippin, 1984; Cuthrell and Biegler, 1989; Diwekar, 1995; Sundaram and Evans, 1993; Mujtaba and Macchietto, 1993). However, relatively little has been published on the dynamic optimization of an entire batch process, in spite of the fact that Barrera (1990) demonstrated that optimizing isolated unit operations cannot take advantage of the significant tradeoffs that may exist between processing operations. Both simple algebraic and detailed dynamic models have been employed to predict the effects of recipe modifications on the performance of the entire process, and both rigorous and ad hoc procedures have been used to solve the resulting models. These approaches address the continuous decisions defining the operating policies of the tasks, yet none are able to cope with the discrete decisions related to the equipment allocation at the same time.

Algebraic Performance Models

Tricoire (1992) considered the planning and design of multiproduct batch polymer processes. He argued that the detailed operating decisions could not be considered during the design of the overall plant design, particularly for polymer processes in which the temperature policy and initiator feed rate offer a huge number of possible operating scenarios. He identified key parameters associated with the performance of the tasks and selected these as the decision variables for the plant design, and provided correlations to relate these variables to the size factor, batch size, and cycle time for the tasks. The resulting design problem was solved using a simulated annealing algorithm to improve the operation of the process. Improvements over designs in which the operating conditions were fixed were gained through the application of the procedure. His research demonstrates the benefits that can be obtained through operating policy modifications, even when approximate models are employed.

Salomone and Iribarren (1992) demonstrate that some batch processing operations can be approximated using algebraic models. Size factors and processing times are expressed as explicit posynomial functions of certain key operating parameters through symbolic rearrangement of the algebraic model. Key operating parameters are selected and manipulated to optimize a heuristic design target suggested by Yeh and Reklaitis (1987). The size factors and processing time functions that optimize the target are then used as the data for the posynomial model for plant design formulated by Grossmann and Sargent (1979). The resulting design incorporates operating decisions and accounts for the interaction between task performance and plant scheduling, but the operating parameters are determined before the plant design problem is solved.

Montagna *et al.* (1994) demonstrate that the optimization of the size factors and cycle times can be conducted at the same time that the optimal unit sizes are determined, and show that the optimal operating conditions differ for a given product depending on whether it is produced in a dedicated facility or as one of a slate of products manufactured in a multiproduct facility. They employ the algebraic models used by Salomone and Iribarren (1992) and add estimates for the utility costs to the objective. They embed the equations defining the size factors and cycle times as constraints in the posynomial model for the optimal plant design, forming a general (non-convex) nonlinear program. They assume that the discrete decisions relating to the plant design, such as the number of equipment items in parallel, the storage policy, and the task to stage assignment, are made either before the optimization is undertaken or that they are determined in an outer optimization loop. They suggest that heuristic procedures (Tan *et al.*, 1993) may be used to aid in the calculation of the optimal values for the discrete decisions.

These two approaches have several drawbacks. Even though these models have been solved systematically, the usefulness of the resulting solution is called into question because the complex time-dependent behavior of the processing tasks has been replaced with algebraic approximations. In addition, the symbolic rearrangement required to generate explicit expressions for the size factors may not be possible. Although the Montagna *et al.* (1994) formulation does not require symbolic rearrangement, the optimization is likely to suffer from the nonconvexities in the feasible region. Furthermore, if the discrete decisions are made in an outer optimization loop, well known MINLP decomposition techniques cannot be employed because the nonlinear models are nonconvex (Sahinidis and Grossmann, 1991; Bagajewicz and Manousiouthakis, 1991). Thus, the outer loop iteration will either be entirely heuristic or will be doomed to total enumeration of the discrete space.

Detailed Dynamic Performance Models

Barrera (1989; 1990) demonstrated that detailed dynamic models could be employed to optimize the performance of a batch process. A set of operating parameters were identified as the decision variables, and the optimization of the process performance for a given allocation of equipment was posed as a nonlinear program; the solution of the dynamic models was considered as part of the objective function evaluation (essentially a control vector parameterization decomposition). A sequential quadratic programming algorithm was used to solve the resulting problem. The processes examined contained no recycles, so dynamic models of the tasks were solved sequentially in order to evaluate the process performance. Operating constraints related to product purity and temperature were included as constraints in the NLP. Barrera included this performance optimization as part of an ad hoc iterative procedure to determine the operating policies and equipment allocation required for process development.

Wilson (1987) determined the optimal performance of a reactive batch distillation process. The process consisted of a reaction step and a separation step that could be conducted in the same vessel. Simultaneous reaction and separation allowed purification of the product during the reaction step which enhanced the reaction performance. Both the capital cost of the reactive distillation unit and the operating and raw material costs of the process were considered. The process was modeled by a set of differential equations which were solved using a Runga-Kutta integrator. The optimal operating conditions and column size were determined through an ad hoc manual search over the key variables. His work demonstrates the benefits of simulation during the design of both the process and the plant, but the simple, one-unit process considered avoids the complications caused by the interactions between different processing stages.

Salomone *et al.* (1994) extend their earlier work on the batch plant design problem to enable the use of dynamic models. They developed an iterative algorithm which utilizes dynamic models to calculate the parameters for the posynomial models used to minimize the annualized investment and operating cost during equipment sizing. The formulation results in a nonlinear program in which a subset of the operating parameters are selected as the decision variables; the authors do not state what procedure is used to update the decision variables or how the updates are determined. During what would normally be a function evaluation, the DAE models of each task are solved, and any material recycles are converged. It is assumed that product specifications can be met at the assigned values of the decision variables. Next size factors and expressions for the processing times are determined from the simulation results using symbolic manipulation. With this information, the posynomial model is solved to provide both the optimal equipment sizes and the value of the objective function for these operating conditions. The iteration strategy they propose is very similar to the process outside–structure inside (POSI)[1] iteration proposed by Barrera (1990); the structure subproblem used to optimize the equipment allocation within process development has merely been replaced by the posynomial model used to select the optimal equipment sizes for the plant design. The optimization they propose cannot deal with values of the decision variables that are unable to satisfy the product specifications, and the method cannot handle path constraints.

Bhatia and Biegler (1996) considered the design of a batch plant in which the equipment sizes and the operating policies of the tasks were optimized using dynamic optimization. They considered a sequence of processing tasks without material recycles operating in either the zero-wait or unlimited intermediate storage mode of operation. The tasks were modeled using simple differential algebraic models of the

[1] See figure 2-1.

tasks; for instance, they employed a shortcut distillation model based on the Fenske, Underwood, and Gilliland correlations. The scheduling of the units is determined by calculating the limiting batch size and cycle time of the processing trains. They formulated the optimal process design as a dynamic optimization problem in which the operating policies of the tasks and the equipment sizes were determined. The problem was solved by transforming the dynamic optimization to an NLP through orthogonal collocation on finite elements (Logsdon and Biegler, 1989). Their approach demonstrates the ability to employ dynamic models directly within the optimization procedure, but the size of the models employed does not reflect the level of detail often required. Extension of the method to larger process models will depend on the ability of the NLP code to handle large process models. In addition, application of this method requires the user to be able to provide enough finite elements to maintain the accuracy of the solution of the DAEs, and it is not clear how to determine the required number of elements beforehand. See section 6.3.2 for a discussion of the merits and drawbacks of the collocation approach for the solution of dynamic optimization problems. Furthermore, incorporating discrete decisions into their formulation leads to the formation of a large nonconvex MINLP.

Charalambides *et al.* (1993) proposed to determine the optimal operating policies and equipment sizes via the solution of a multistage dynamic optimization problem employing detailed differential-algebraic models of the tasks. They demonstrated that a control vector parameterization approach (Kraft, 1985; Vassiliadis, 1993) could be used to convert the dynamic optimization to a finite dimensional problem, allowing the application of conventional gradient based nonlinear programming techniques. In addition, representing the process as a state task network and defining the material states in terms of time-invariant optimization parameters removes all direct interactions between the processing tasks. The decoupled task models and corresponding sensitivity equations can be integrated in isolation, permitting parallelization of the time-consuming integrations. Charalambides *et al.* (1995a; 1995b; 1996) applied this technique to several examples, demonstrating that the formulations could be solved in times that are reasonable for design calculations. However, their technique is limited to continuous dynamic models and cannot employ the hybrid discrete/continuous dynamic models that we have argued are required to represent many batch process operations, particularly those in which phases appear and disappear during the operation of the task. Extending their technique requires the ability to transfer the parametric sensitivities across implicit discontinuities, as formulated by Barton (Barton, 1996).

2.1.3 Coupling the Structure and Performance Subproblems

A seemingly natural extension of the work of Montagna *et al.* (1994) would employ the algebraic performance models within a mixed-integer nonlinear programming (MINLP) framework. Unfortunately, nonconvexities in the model make the application of conventional MINLP techniques invalid (Sahinidis and Grossmann, 1991; Bagajewicz and Manousiouthakis, 1991), since the bounding properties of the relaxed problem cannot be achieved. While an analogy between these algebraic models and the screening models we present is evident, the models of Montagna *et al.* (1994) do not possess provable bounding properties that can be exploited to prune discrete alternatives.

In contrast, Barrera proposed a method to solve the process development problem with detailed dynamic models via a decomposition approach. His approach requires iterating between the performance and structure subproblems, fixing the variables used in one subproblem when the other subproblem is solved; the performance is optimized for a given structure, and the structure is optimized for fixed operating policies. Barrera used an SQP algorithm to solve the performance subproblem (solving the DAEs during each function evaluation), a local search method to solve the structure subproblem, and an ad-hoc procedure to iterate between the two subproblems. Using this procedure he clearly demonstrated the benefits that could be gained by considering the optimization of both resource allocation and operating policies simultaneously. The strategy is implemented using a nested iteration, and the two nesting strategies shown in figure 2-1 were examined. He found that the choice of nesting strategy had a significant impact on the solution time because the performance subproblem took far longer to solve than the structure subproblem. Therefore, the POSI strategy, in which the performance subproblem is solved in the outer loop and the faster local search algorithm is employed on the inner loop, was found to be more efficient. The outer iteration loop was continued until little improvement in the objective function was observed.

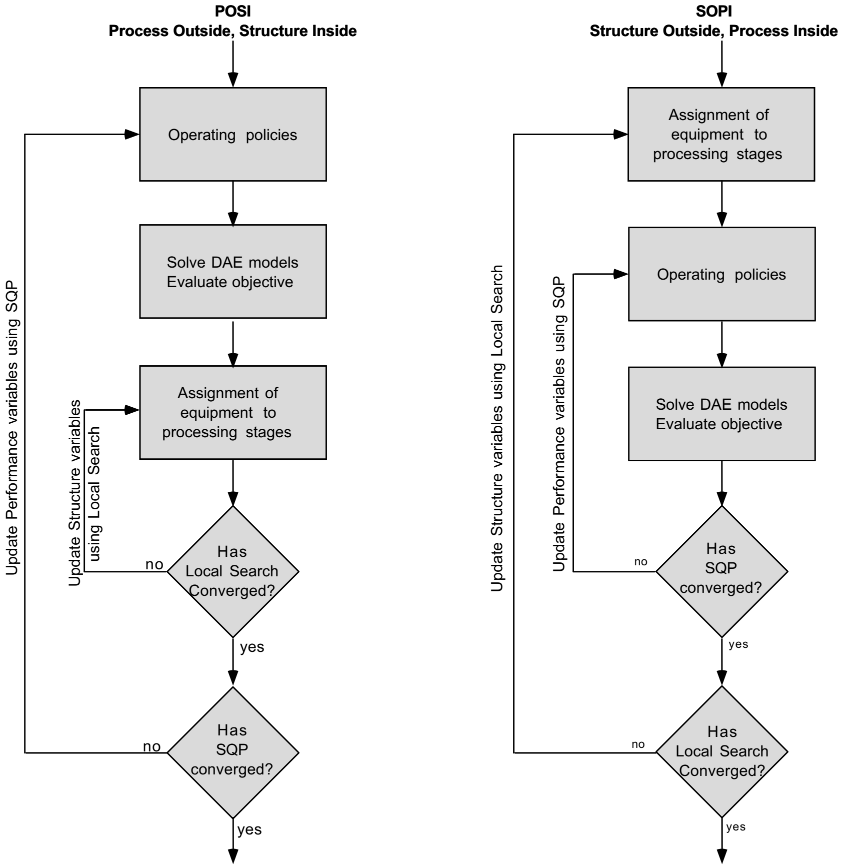
POSI
Process Outside, Structure Inside
Operating policies
Solve DAE models
Evaluate objective
Assignment of equipment to processing stages
Has Local Search Converged?
no
yes
Has SQP converged?
no
yes
Update Performance variables using SQP
Update Structure variables using Local Search
SOPI
Structure Outside, Process Inside
Assignment of equipment to processing stages
Operating policies
Solve DAE models
Evaluate objective
Has SQP converged?
no
yes
Has Local Search Converged?
no
yes
Update Structure variables using Local Search
Update Performance variables using SQP

Figure 2-1: The two nesting strategies for the performance and structure subproblems investigated by Barrera (1990).

Barrera's approach highlights the need to improve the strategies to iterate between the two subproblems when a decomposition approach is employed; in particular, discrete alternatives cannot be eliminated from consideration, because neither subproblem provides a lower bound on the overall objective. More importantly, a metric for assessing the potential benefits of continued optimization is sorely needed.

Charalambides *et al.* (1993) also postulated a multistage dynamic optimization problem containing integer variables for the solution of the batch plant design problem. They noted that applying control vector parameterization and treating the integer variables as time-invariant parameters results in a nonconvex MINLP optimization problem. No solution procedures or examples with discrete decisions have been presented in the literature to date.

2.2 Applying Screening Models to Process Development

Screening models for process development yield a lower bound on the cost of manufacture by considering changes to the process structure, the operation of the tasks, and the allocation of equipment simultaneously. The models embody a convex underestimate of the objective and a convex overestimate of the feasible region. The screening models enable a simultaneous approach to the process development problem shown in figure 2-2 that contrasts the sequential and iterative approaches shown in figures 1-1 and 1-2. The drawback is that the models do not consider the detailed operation of the tasks, so the model solutions do not correspond to designs that can be implemented directly. Instead, the screening model provides targets for the detailed design of the actual process. These screening models are also capable of performing aspects of the process synthesis. In addition, the screening model can be used to enhance the application of existing approaches, or as the basis for a rigorous decomposition strategy to address the process development problem as a mixed-integer dynamic optimization problem (Allgor and Barton, 1997b).

The lower bounding property possessed by these models motivates the term 'screening model', since the bound can be used to prune or screen discrete alternatives that cannot lead to the optimal solution, avoiding the need for total enumeration of the discrete decision space. For example, Daichendt and Grossmann (1994a; 1994b) employed screening models to prune the branch and bound tree in order to improve the efficiency of a MINLP algorithm used for heat exchanger network optimization. For batch process development, screening models can be used in a similar fashion. Given

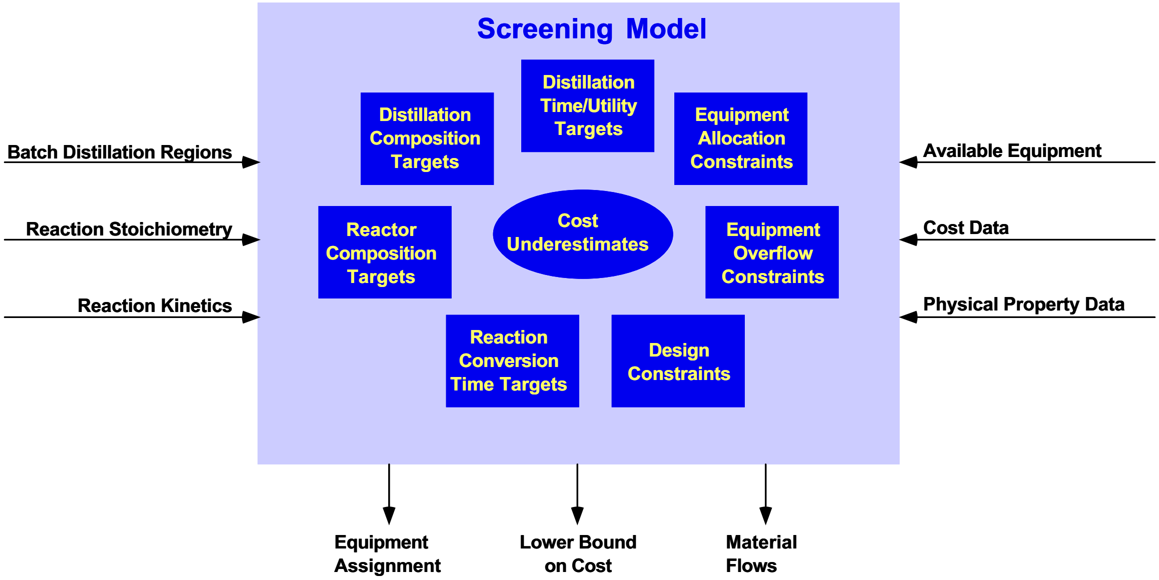
Screening Model
Distillation Time/Utility Targets
Distillation Composition Targets
Equipment Allocation Constraints
Reactor Composition Targets
Cost Underestimates
Equipment Overflow Constraints
Reaction Conversion Time Targets
Design Constraints
Batch Distillation Regions
Reaction Stoichiometry
Reaction Kinetics
Available Equipment
Cost Data
Physical Property Data
Equipment Assignment
Lower Bound on Cost
Material Flows

Figure 2-2: Schematic of the information provided to and produced by the screening formulations.

an initial ‘base case’ design, these formulations can be used to prune all discrete alternatives with greater cost than the base case, yielding a set of candidate structures that offer the potential for improved performance. The performance subproblem can then be solved for each of these candidate discrete alternatives using dynamic optimization. Such a procedure is capable of determining the best design that can be found using the available dynamic optimization algorithms, without performing total enumeration of the discrete alternatives. However, global optimality cannot be guaranteed because the dynamic optimization is not guaranteed to find the global optimum; in fact, most dynamic optimization problems exhibit multi-modal behavior almost pathologically (Banga and Seider, 1995).

The design targets provided by the screening formulations can also be used to enhance iterative approaches for batch process development. Since Barrera’s approach is strictly a local search technique, the resulting solution could be far from the global optimum, yet the approach has no way of measuring or estimating the distance to the optimum. On the other hand, the solution of the screening model provides an underestimate of the global optimum that can be used to estimate the quality of the design obtained by the iterative procedure and to assess the potential benefits of continued optimization. If significant improvements are possible, an iterative procedure can be repeated, starting from a different initial point. The solution of the screening model provides a reasonable candidate for the initial point of continued optimization using the iterative procedure.

Screening models also facilitate the application of multistage dynamic optimization algorithms to the optimization of the operating policies for a batch manufacturing process performed in dedicated equipment items. Multistage dynamic optimization decouples the tasks using the material states (Charalambides *et al.*, 1993), yet it requires a priori definition of the state task network (STN), initial guesses for the states (treated as time invariant parameters), and the definition and initialization of the admissible functions for the control variables. The solution of a screening model facilitates definition and initialization of all these quantities.

Dynamic optimization requires definition of the STN before the optimization is attempted. This implies that the number of states included in the process and the way in which they are connected to the tasks must be defined beforehand, defining the recycle structure of the process. For example, each distillation cut, including off cuts, requires a separate state node in the STN, so the number of cuts permitted for the distillation tasks is also represented in the definition of the STN. The solution of the screening model defines the number of cuts that would be required if perfect splits could be achieved and a feasible recycle structure utilizing the sharp splits. The actual number of cuts provided in the STN must account for off cuts as well, but should reflect the information gathered from the solution of the screening model. Embedding redundant process structures within the STN, such as unnecessary distillation cuts, may create several problems for the dynamic optimization algorithm. First, this will increase the multi-modal character of the optimization problem. For example, consider the dynamic optimization of the solution of the screening model for the first superstructure of the case study considered in chapter 4 shown in figure 4-3. Since the system contains six components, we would expect that we might require five overhead distillation cuts if we defined a general state task network for the process. However, the solution of the screening model indicates that only two overhead cuts are required for the first distillation task and only one for the second. Thus, we can pose a STN for the dynamic optimization based on the information collected from the solution of the screening model that contains fewer overhead distillation cuts such as that shown in figure 2-5.[2] Note that we could also augment the STN shown in figure 2-5 to include off cuts. If we had included five overhead cuts with each of the distillation tasks and permitted each of these cuts to be sent to any of the other tasks, we would have a superstructure for the dynamic optimization that is highly redundant. If only two cuts are required, but five are allowed, then the optimal solution could contain any two of the four cuts (or could take fractions of the two required cuts). Similarly, incorporating tasks that are not performed in the STN and relying upon the optimization to remove them by setting the flow rates into the task to zero may cause problems for the optimization. The model of the task may not be defined in the absence of material, and even if no material is present, sensitivities are still required for the controls related to these tasks. Including unnecessary tasks can also lead to redundancy. For instance if two reaction tasks are allowed but only one is required, then the active reaction task could be either the first or the second reaction task. Progress in dynamic optimization techniques may help mitigate these difficulties, but current algorithms are likely to be more reliable if they are presented with a reasonable problem and given an initial guess in the vicinity of a unique local optima.

Since, in general, the dynamic optimization can find a local optimum at best, the starting point will affect the solution that is obtained. Successful application of multistage dynamic optimization techniques requires good initial guesses for the material states and for the control profiles at the very least. Initial guesses for the intermediate material states can be assigned using the solution of the cyclic steady state mass balances provided by the screening model. The screening model will provide compositions of the intermediate states that are consistent with the structure of the STN and expected to be near the optimal values. Since the performance of the

[2] The tanks represent the state nodes of the STN, and they are characterized by time invariant optimization parameters.

distillation changes qualitatively depending on the location of the feed with respect to the batch distillation boundaries, the optimization will almost certainly have great difficulty crossing from one distillation region to another. For example, consider a feed located in batch distillation region three of figure 4-1. If we expect the first cut from the distillation to contain mostly B and possibly some A (the lightest components in the system that both happen to be reactants), we may want to recycle this cut to the reactor. We would construct a STN that embeds this possibility, and we provide an initial guess to the dynamic optimization for the composition of this state that is mostly B. However, if the dynamic optimization moves the feed to the distillation column into region II, the first cut from the column will have a composition close to that of P-W_1 instead of B. This will cause a large violation of the optimization constraints that equate the composition of the recycled distillation cut to the feed to the reaction task. Thus, we need to consider the active batch distillation region when constructing the STN, even though the optimization could theoretically move from one region to another. More importantly, this observation demonstrates that the structure of the STN must be consistent with the initial guess provided for the compositions. Starting with good initial values for the parameters is also likely to decrease the time required to obtain a solution of the dynamic optimization. However, the dynamic optimization will contain more variables than the screening models, so a strategy to approximate the quantities not explicitly defined by the screening formulation will be required.

Many of the benefits accruing from the use of screening models in conjunction with dynamic optimization are due to the synthesis features of the screening formulations. The dynamic optimization only addresses the design aspects of the process recipe, yet the recipe comprises both design and synthesis information. Screening models have the ability to address aspects of the process synthesis not considered by previous batch process design procedures. Although the reaction pathways and processing steps employed at the bench scale need not remain fixed during the process development, in many cases sufficient information is not available to predict the effect of synthesis changes without resorting to detailed bench scale experimentation. For instance, screening models require reaction stoichiometry and kinetic information, so the models can choose between several alternative reaction pathways embedded within the superstructure, but could not invent new pathways. Similarly, decisions involving the selection of reagents and solvents from a list of candidates (see Modi *et al.* (1996) for example) can be determined during the solution of the screening model. The superstructure provided by the screening model for reaction/distillation networks allows for the appearance and disappearance of both reaction and distillation tasks. Thus, the screening model defines the choice of reactants and solvents for the process, selects the tasks that will be performed, and defines the recycle structure for the process — tasks that are traditionally considered the domain of the process synthesis. In addition, the screening models can distinguish between different process structures. This ability is illustrated by the case studies considered in chapters 4 and 5; in both cases the screening model selects a processing structure that differs from the process structure employed by the chemist at the bench scale.

Screening models also enable the derivation of a rigorous algorithm to address the mixed-integer dynamic optimization formulation of the batch process development problem. The lower bound provided by the screening model is the key to generating an iteration that can rigorously prune portions of the discrete space. A rigorous iteration procedure that guarantees improvement of the solution and potentially avoids explicit enumeration of the entire discrete decision space is derived by iterating between the screening model and dynamic optimization of the operating policies (Allgor and Barton, 1997b); this is discussed in detail in section 2.4 and in chapter 9.

2.3 Scope of Development Problems Considered

The general form of the batch process development problem is too complicated to propose a systematic model-based solution procedure at present. For example, dynamic models for batch reaction and distillation tasks are readily available, but for many processes, especially those involving biological transformations or other unit operations most commonly encountered in batch processes (e.g., crystallization, drying, extraction), dynamic models capable of accurately predicting the performance of the task in terms of the operating variables are not yet available. In addition, the interactions between the processing operations and the manufacturing facility require that fairly detailed information about the plant is provided.

Therefore, this thesis focuses on a subset of these problems that can benefit from detailed modeling of the tasks. Future research may allow some of the following restrictions to be relaxed:

- Only unit operations that can be modeled with state of the art process modeling technology will be considered. This implies that only limited effects of scale can be considered. In fact, the screening models further restrict the class of processes considered to networks of reaction and distillation tasks.

- Sufficient experimental and physical property data is available, or can be obtained and/or estimated to describe the system to the required level accuracy.

- Products will be manufactured in campaigns.

- Although it is an important issue, uncertainty in the model parameters will not be considered explicitly in the design; however, sensitivity studies can be conducted.

Since the design of the process defines the interactions between the recipe and the equipment, we examine the way in which both the process recipe and the manufacturing facility are represented for the problems and case studies considered within this research.

The development problem considered within this research considers manufacture within an existing manufacturing facility. Since the plant already exists, we merely need to find a representation that provides sufficient detail for the engineer to ascertain the feasibility of proposed designs. The notion of a *plant superstructure* will be used to represent the processing facility. The superstructure consists of the equipment items, utilities, valves and interconnecting piping, and plant instrumentation available within an existing facility.

The process recipe, on the other hand, requires quite a different representation. The process can be thought of as a sequence of processing tasks and operations which transform the raw materials into desired products and waste materials. A powerful representation of this is provided by the State Task Network (Kondili *et al.*, 1988). Although the state task network has been most frequently associated with discrete time batch scheduling formulations, it is a general representation for the process recipes that is particularly appropriate for the purposes of process development. The STN provides a graphical representation of the process. It is a directed graph composed of two types of nodes — state nodes and task nodes. The task nodes correspond to processing tasks and are just like the nodes in a continuous process flowsheet. However, in the STN the task nodes are not associated with a particular item of equipment. The state nodes represent material (e.g., raw materials, intermediates, and products) in a specific thermodynamic state. Every arc in the digraph connects a node of one type, state or task, to a node of the other. The networks can be arranged in a general fashion, but if two arcs are incident upon the same state node, they must carry material in exactly the same thermodynamic state. The STN provides a convenient framework in which to express the equipment assignment constraints (i.e., scheduling). Moreover, the STN provides a general abstract representation of the recipe that can be used to describe the process in terms of parameters that can be determined by automatic search procedures such as dynamic optimization. Charalambides (1996) devotes an entire chapter of his thesis to the representation of process recipes using the state task network.

Figures 2-3 and 2-4 give examples of the representations employed for both the plant and the process recipe, respectively. The figures depict a reaction task that transforms two raw materials into an intermediate. The representation of the process is not tied to particular equipment items, and the plant is not reserved for a particular product. Note, however, that the superstructure of the plant limits the operating procedures that may be considered for implementation of the process. For instance, the first feed tank has a feed pump for each reactor, but the second tank has only one feed pump. This limits the feed policies that may be considered. The operating

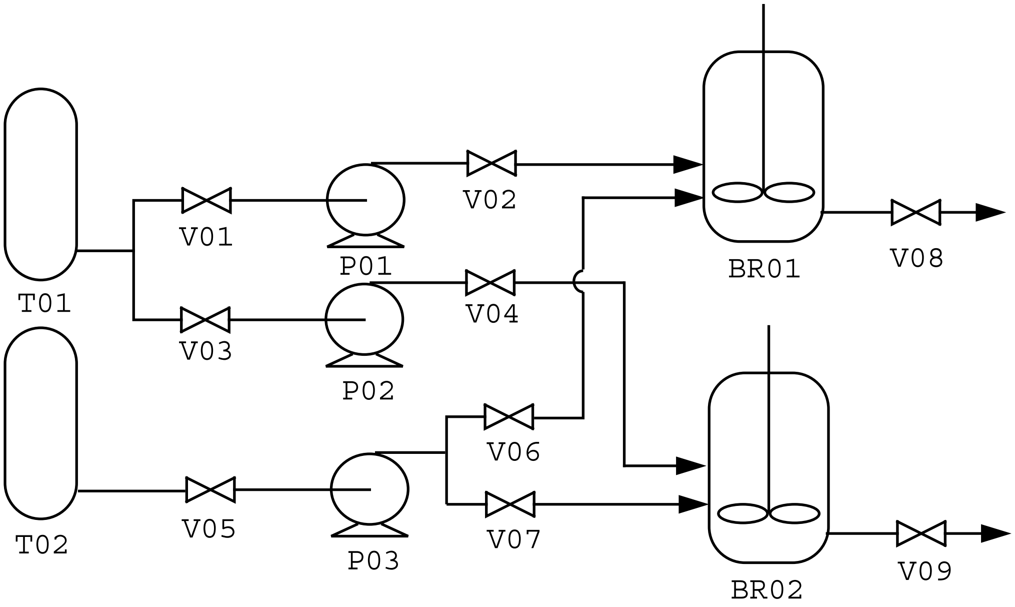

Figure 2-3: Plant Superstructure for Batch Reactor

limitations imposed by the plant superstructure must be considered during the process development.

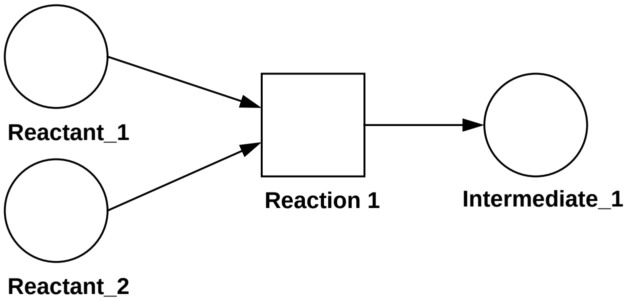

Figure 2-4: State Task Network for Batch Reaction

In general, alternative processing structures (i.e., the selection of batch distillation or an absorption desorption process (Charalambides, 1996)) can be represented within the framework of the state task network. However, if alternative processing structures are included, then the design methodology must be capable of deciding between the alternatives. For this reason, two different abstractions for the structure of the process recipe are used within the decomposition strategy for batch process development described in the next section. The process superstructure employed by the screening models, which is a restricted form of state task network, provides alternative processing configurations. The screening models are able to select between these alternatives as demonstrated by the case studies in chapters 4 and 5. However, current dynamic optimization techniques cannot select between alternative processing configurations in most cases, so the state task networks representing the process recipe employed during the application of the dynamic optimization do not contain alternative processing structures.

The reason that the dynamic optimization techniques cannot choose between alternative processing configurations is that different equations are typically required to represent the processing operations when they are performed and when they are idle. For example, when a distillation column is operating normally the holdup of material on the trays and in the reboiler are nonzero and the intensive properties of the system are well-defined. However, if the column remains idle, the holdup of material is zero, and the intensive properties are not defined by the typical relationships. Combined discrete/continuous modeling languages permit models that consider these two cases using separate sets of equations to represent each situation, switching between them when the appropriate conditions are satisfied (Barton, 1992). However, current dynamic optimization methods cannot handle situations when the model equations can change implicitly. Note that this situation may soon change; in fact, recent theoretical advances defining the parametric sensitivities across implicit discontinuities (i.e., state events) permit gradient based dynamic optimization of general hybrid discrete/continuous models using control vector parameterization (Barton, 1996). In either case, the dynamic optimization problems representing the performance subproblem employ a STN that contains the subset of the processing alternatives that has been defined by the solution of the screening model.

The flexibility with which equipment can be assigned to processing tasks within the screening models is similar to the equipment configurations considered in the batch process scheduling literature. The case studies assume that equipment units are chosen from the inventory of equipment and reserved for the manufacture of the desired product until the end of the campaign. At the start of the campaign, a pipefitter makes the necessary connections between the processing equipment; these connections remain in place until the campaign has been completed. The case studies demonstrate that the screening models can consider this level of flexibility with respect to the equipment assignment. However, the equipment configurations available within most manufacturing facilities are far more restrictive that those that have been allowed within the screening models. Although some toll manufacturers do in fact operate in this fashion, it is only practical to connect vessels that are situated in the same vicinity or vessels that can be easily moved. Many large specialty chemical and pharmaceutical manufacturers have far more structured and restricted equipment configurations. The processing equipment within their facilities is typically housed in a number of buildings that each contain several production areas. Each production area may contain 3 to 4 production bays. The production bays contain a variety of equipment such as reactors, filters, and storage vessels of similar size. Several bays may share some common items of equipment for drying and solvent switch operations. Large facilities may have about 100 production areas on a given site. However, a much smaller number of these may be suitable for a particular process. For example, some are reserved for high pressure operation, some for atmospheric operation or slightly above, and other bays may not possess the equipment required for some processing steps. Thus, for a particular set of reaction steps a much smaller number of bays may be appropriate. Many of these facilities also separate the solvent recovery operations from the reaction steps. All of these restrictions can be represented as additional constraints in the formulation presented in chapter 3. In summary, the combinatorial aspects of the equipment allocation considered within this research are more than adequate to represent the equipment options available to most manufacturers. In fact, in many cases, the flexibility considered here is far greater than the situation facing many manufacturers. In particular, note that the scheduling of these processes is far more restricted that the scheduling of blending and formulation operations, commonly examined in the scheduling literature, where the combinatorial complexity can be many orders of magnitude greater, but where detailed dynamic modeling is not likely to lead to dramatic improvements in the process efficiency (even if adequate models exist).

2.4 Decomposition Algorithm for Batch Process Development

The ability of the screening model to consider the discrete and the dynamic operating decisions simultaneously and solve the resulting model to guaranteed global optimality permits the derivation of a rigorous decomposition algorithm for batch process development. The algorithm employs mathematical models of the processing tasks at two levels of detail: algebraic screening models that provide rigorous lower bounds on the production cost, and detailed dynamic models that accurately predict the process performance.

The extension of traditional mixed-integer nonlinear programming decomposition methods (Geoffrion, 1972; Duran and Grossmann, 1986) to batch process development and to other mixed time invariant integer dynamic optimization problems is thwarted by the inability to derive a valid Master problem using information provided by the primal, among other problems (Allgor and Barton, 1997b). However, the screening model's lower bounding property permits it to be employed as part of a decomposition strategy for the solution of the mixed-integer dynamic optimization. This algorithm is discussed in chapter 9.

The algorithm decomposes the original process development problem into two subproblems. The solution of the first, the screening model, provides a lower bound on the cost of future solutions. The second subproblem is the performance subproblem which is formulated as a dynamic optimization problem in which the discrete decision variables in the original problem take the values determined by the solution of the corresponding screening model; its solution yields a feasible detailed design.

The screening model provides information that is either required or beneficial for the formulation and solution of the dynamic optimization problem that corresponds to the performance subproblem given the allocation of the plant resources defined by the solution of the screening model. The solution of the screening model provides:

1. A definition of the processing structure, defining what operations should be included and what operations are not required.

2. An assignment of equipment items to the tasks that are performed. These equipment items are selected from the manufacturing facility's inventory, and dedicated to a particular task or set of sequential reaction tasks for the duration of the campaign.

3. Information indicating which batch distillation regions are active. Since the active batch distillation region is represented using a discrete variable, qualitative changes to the performance of the distillation column resulting from feeds in different regions can be easily identified.

4. The number of distillation cuts required under ideal conditions. While more cuts may be required in the detailed design, the number of cuts given by the screening model provides information that can be employed to decide how many cuts and off cuts should be considered during the dynamic optimization.

5. Definition of the basic structure of the state task network defining the process for these values of the discrete decision variables.

6. Initial values for the compositions of the state nodes within the STN described above. The state nodes represent either recycled material or material that decouples the dynamic interactions between processing tasks (i.e., material that leaves the reaction task and is fed to the distillation task at the start of the next batch). The values of these states defined by the screening model may not be feasible for the dynamic optimization, but they should provide a good initial guess for the optimal values.

Next, we examine how this information facilitates the formulation and solution of the corresponding dynamic optimization problem. The solution of the screening model for the first case study shown in figure 4-3 will be used to demonstrate the points.

Note that a mixed time invariant integer dynamic optimization formulation of this same example is given in section 9.5.

Since the performance of a processing task may depend on both the chosen operating policies and the characteristics of the equipment in which it is carried out, the performance subproblem requires that the equipment items assigned to each processing task are known. In this algorithm, these assignments are fixed by the solution of the corresponding screening model, so the appropriate dynamic model can be selected for each task when formulating the dynamic optimization. In addition, the inequality path constraints may depend on the equipment assigned to the processing task (e.g., equipment overflow constraints, maximum vapor rate constraints, etc.), so the equipment assignment must be known before the appropriate dynamic optimization can be solved.

In order to formulate the dynamic optimization subproblem, the state task network for the process must be defined. We could choose to include many states and tasks that may not be required, but this will lead to redundancy in the solutions that may be obtained. Instead, we choose to employ the information provided by the solution of the screening model to construct a state task network for the process that reduces the size of the resulting dynamic optimization by eliminating redundant processing tasks; removing redundant processing tasks also improves the performance of the optimization algorithm. The key pieces of information that are required to construct an appropriate state task network are the number of tasks that are included in the processing network, the number of cuts (and potential off cuts) taken from each of the separation tasks, and the recycle of material within the process indicating where the material produced by one task is next used. Once these decisions have been made, the processing structure is determined. Comparing figure 2-5 to figure 9-3 clearly shows that the process structure defined by the solution of the screening model is much simpler than the process structure that allows for all the cuts that might be required in each of the separation tasks. In fact, the screening model predicts that only two overhead cuts are required for the first distillation and only one is required for the second. Without this knowledge, we would allow for five overhead cuts in the process structure because the process contains six components. Furthermore, the recycle structure of the process is defined by the screening model, simplifying the material balances around the tanks defining the material states. Using the process structure defined by the screening model allows us to eliminate redundancy in the definition of the process structure which should permit the dynamic optimization algorithms to perform better, since all of the optimization parameters should affect the objective value. In contrast, including cuts that are not required will lead to multiple solutions with the same objective value, which will probably degrade the performance of the optimization algorithm.

The dynamic optimization formulation of the performance subproblem solves for both the operating profiles of the processing tasks and the values characterizing the states in the STN simultaneously. In the example shown here, the temperature profile in the reactor, the reflux ratio of the columns, and the split fraction determining the distribution of flow between the two overhead cuts on the second column are treated as the controls. The composition and amount of material in each of the state nodes generated for each batch is also determined; in figure 2-5 the state nodes are represented using storage tanks that hold the material. Material transfers occurring at the beginning and end of a task are represented using the gray lines with larger dashes, and the constraints depicting the transfer of material from one task to the next are shown using small black dashed lines. The solid lines represent material transfers during the task. Note that this picture assumes that both the reactors and columns are operating in batch rather than fed batch mode. The per unit manufacturing cost of in-spec product is minimized during the solution of the performance subproblem.

By comparing the STN shown in figure 2-5 to that shown in figure 9-3, we observe that we are only considering a subset of the potential processing structures. We recognize that this may exclude better solutions, but the dynamic optimization algorithms cannot guarantee convergence to a global optimum. This implies that the initial guess provided to our dynamic optimization procedure may have a greater impact on the quality of the solution obtained than the number of processing structures embedded in the STN. The screening model provides initial guesses for all of

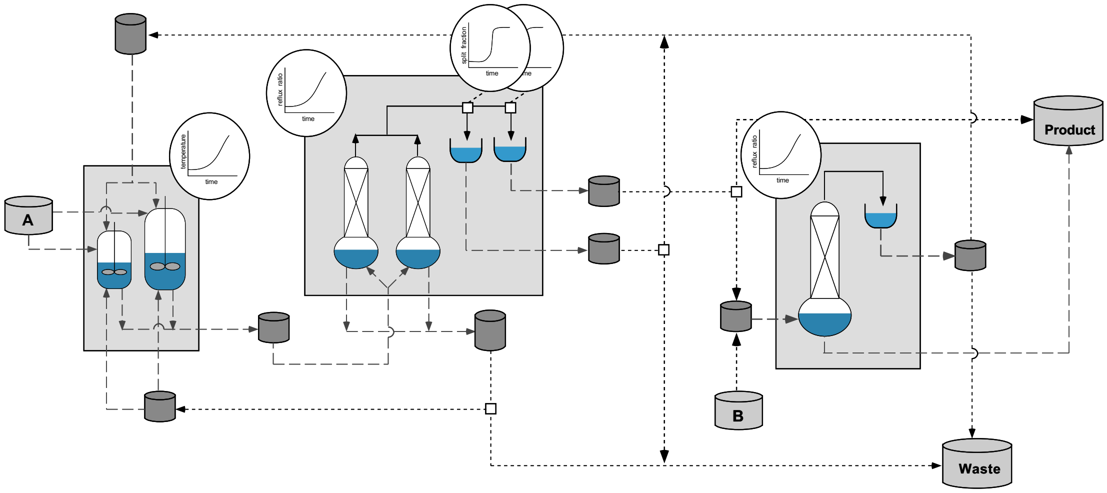

Figure 2-5: The state task network for dynamic optimization of the process development example from chapter 4. This corresponds to the screening model solution obtained from the first process superstructure.

the material states appearing in the process structure defined by the solution of the screening model. Although the detailed dynamic models may not be able to achieve the material compositions predicted by the screening model, the values predicted by the screening model are expected to be near an optimal solution. Therefore, using the solution of the screening model as the initial guess for the dynamic optimization may actually enable the dynamic optimization to find a better solution. In addition, since the material recycles given by the screening model satisfy the cyclic steady state constraints, the dynamic optimization may be able to determine a solution in fewer iterations.

Another benefit provided by the this iteration procedure is the fact that aspects of the continuous behavior that are known to lead to the multi-modal character of the dynamic optimization are treated as discrete decisions in the screening model. For instance, the active batch distillation region is identified during the solution of the screening model. While the dynamic optimization algorithm can move the feed from one region to another during the optimization, the optimization must also satisfy the constraints on the parameters defining the material states. Since moving the feed from one region to another can change the qualitative behavior of the distillation, the composition of the material in the accumulator at the end of the distillation task may differ wildly from the parameters corresponding to material in the tank fed by the accumulator. Since the optimization contains constraints that require that the composition of the material in tank representing the state node is equal to the material in the accumulator at the end of the task, the large difference in composition will result in a large violation of this constraint. The NLP solver will most likely force the distillation feed back into the original batch distillation to reduce this constraint violation. In our algorithm, the dynamic optimization will investigate processes with feeds in other batch distillation regions, which may also result in different process structures, during the solution of other instances of the performance subproblem.

The integer cuts added to the screening model at every iteration ensure that previously examined discrete alternatives are not revisited. We treat the inclusion or exclusion of tasks, the assignment of equipment to particular tasks, and the active batch distillation region as the discrete variables defining the structure of the process. The performance subproblem is solved for each of these discrete alternatives until the termination criterion of the algorithm is satisfied. Although we could have chosen to regard only the assignment of equipment and the inclusion of processing tasks in the definition of the discrete alternatives, we would then rely on the dynamic optimization to find the best local optimum of functions that we know to be multimodal. By defining the discrete alternatives as we have, we account for the some of the qualitative changes to the process performance in the discrete domain, allowing us to determine a local optimum in each of these domains through the solution of a different instance of the performance subproblem.

2.5 Summary

This chapter demonstrates that previous research addressing batch process development cannot simultaneously address the discrete and detailed dynamic design decisions in a rigorous fashion. However, previous researchers have derived techniques capable of handling subproblems encountered during batch process development. These techniques are employed within the design method proposed by this thesis. For example, the decomposition algorithm for batch process development proposed within this thesis utilizes the dynamic optimization techniques developed for the performance subproblem and the type of equipment allocation constraints developed for the plant design problem.

The screening models introduced in this thesis permit the derivation of a rigorous decomposition algorithm capable of addressing both the discrete and continuous decisions without requiring total enumeration of the discrete space. This represents the first rigorous approach to the solution of the batch process development problem with the potential to avoid total enumeration of the discrete space. The approach couples insight-based targeting models with gradient based dynamic optimization algorithms. In addition, the screening models can be employed to enhance the application of existing design methods. The derivation of the screening models is discussed in the next chapter.

Chapter 3

Screening Models for Batch Process Development

Batch process development — the design of a process to manufacture a new or modified product within an existing manufacturing facility — is frequently encountered in the specialty chemical and synthetic pharmaceutical industries. Allgor *et al.* (1996) demonstrated the importance of batch process development and stressed the need to develop systematic methodologies that permit the rapid design of efficient batch processes. In order to design an optimal batch process, the optimal recipe and the allocation and scheduling of the plant's resources must be determined simultaneously. This chapter introduces screening models for batch process development that yield a rigorous bound on the cost of the design by considering decisions related to the operation and scheduling of the processing tasks within a single model that can be solved to global optimality.

This chapter introduces the notion of *screening models* for batch process development. Screening models yield a rigorous lower bound on the cost of the process, providing both design targets and a valid way in which to prune or screen discrete alternatives (process structures and equipment configurations) that cannot possibly lead to the optimal solution. These models consider changes to the process structure, the operation of the tasks, and the allocation of equipment simultaneously. In addition, these models embed aspects of the process synthesis not considered in previous research dealing with batch process design. However, they do not provide a detailed process design, so they must be used in conjunction with techniques that consider the dynamics of the process in detail, such as the multi-stage dynamic optimization formulations used to address the performance subproblem (Charalambides, 1996).

In the remainder of this chapter, we discuss the properties that must be satisfied by screening models and derive screening models for batch process development that achieve these properties. In the next section we discuss how information calculated by these models can be employed to enhance existing approaches for batch process development, and how these models facilitate a rigorous decomposition approach for the design of these processes. The application of these models to realistic process development examples is presented in chapters 4 and 5.

3.1 Deriving Screening Models for Reaction/Distillation Networks

The usefulness of screening models hinges on their ability to yield a rigorous lower bound on the cost of the process being developed. To achieve this bounding property, the models must overestimate the feasible region, underestimate the design objective, and consider all of the optimization variables simultaneously. In addition, the optimization procedures used to solve these models must obtain a global minimum. When these conditions are satisfied, the solution of a screening model provides a rigorous lower bound on solution of the original problem.

In order to derive screening models with these properties, constraints related to the equipment allocation and scheduling are expressed in their original form, but the constraints defining the dynamic performance of the processing tasks are relaxed. Algebraic equations representing performance limits replace the differential-algebraic equations describing the task performance, and time averaged material balances are enforced. Therefore, the optimization algorithms used to solve the model must handle both discrete and continuous decision variables, but need not deal with any differential equations. In the remainder of this section, we derive convex models with these properties for the development of batch reaction/distillation networks.

3.1.1 Process Abstraction

We define a superstructure that embeds the synthesis alternatives considered during the solution of the screening model. The process superstructure is represented with a directed graph consisting of state and task nodes. The process is assumed to consist of a sequence of processing trains; each train may contain a reaction and/or a separation task. Stable material is produced by every task. In any train, either task may not exist; note that the reaction tasks must exist if only one reaction pathway is considered and the number of trains equals the number of steps in the reaction pathway. A mixing task prior to each separation task has been included in the superstructure to clarify derivation of the model equations and simplify the notation; these tasks do not require separate equipment items. A diagram of the process superstructure is shown in figure 3-1. In addition, each train of the superstructure is labeled, ordering the reaction steps in the process. Although this ordering has no impact on the superstructure at this level of the hierarchy, it becomes important when the superstructure is refined (see figure 3-5) to consider the purging of recycled streams. The state nodes in this superstructure can be partitioned into two sets, nodes representing the fixed points of a simple distillation process (ρ_1–ρ_{eq} in figure 3-1), whose composition is known before the solution of the model, and nodes leaving the reaction and mixing tasks whose composition is determined during the solution procedure.

The superstructure looks similar to the state task networks (STN) commonly used to represent batch processes for scheduling purposes (Kondili *et al.*, 1988), but it differs from the STN because many of the state nodes in this superstructure do not represent material that can be found in the actual manufacturing process. The product will be manufactured in a campaign with all batches following the same production route, so the process must operate at cyclic steady state. This implies that the arcs in the superstructure correspond to time-averaged material flows. However, these arcs need not correspond to material transfers in the physical process. For

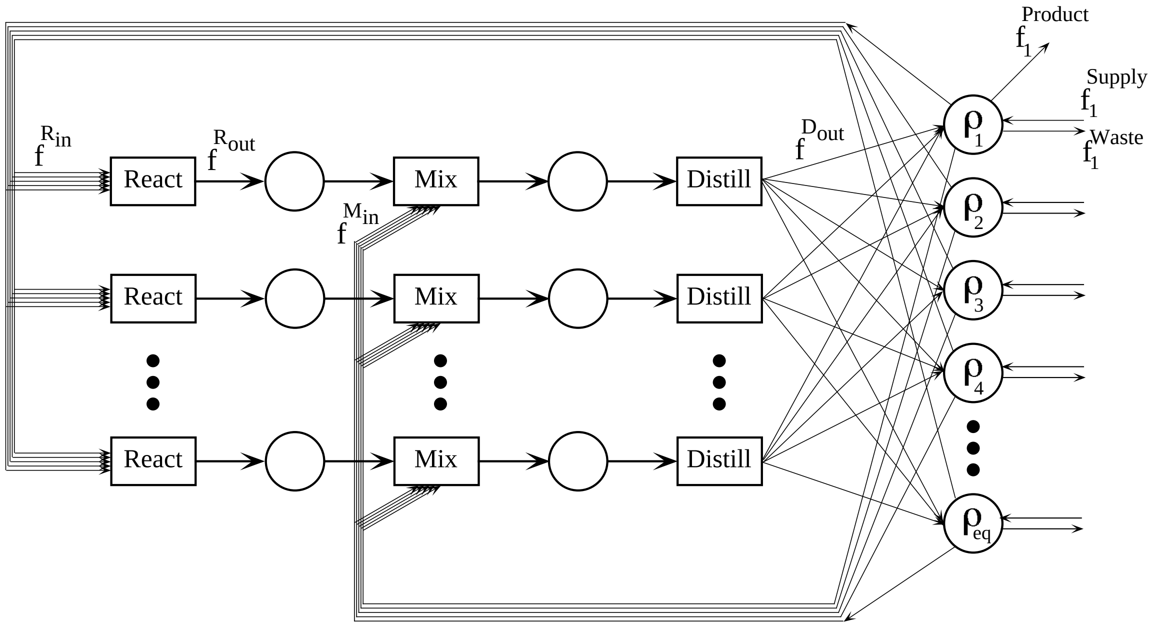

Figure 3-1: Superstructure for networks of reaction and separation tasks.

instance, the targeting procedure used for the distillation tasks permits all feasible separations to be represented in terms of convex combinations of the material sent to each of the equilibrium point nodes. The actual distillation cuts, which may be recycled, processed further, or leave the process as waste or product, are not represented by any single arc of the superstructure.

The time-averaged flows in the superstructure are specified in terms of component molar flow rates; these flows may be specified using either the pure component or fixed point compositions as the basis. The superstructure permits both splitting and mixing of streams, but the splitting of streams leaving state nodes whose composition is not known a priori is not permitted. In order to enforce time-averaged mass balances for this superstructure, models that define the time-averaged flows leaving the tasks in terms of the entering flows and the operating variables are required. To maintain the bounding properties of the formulation, each one of these models must overestimate the region of the composition space that is reachable from a given input specification. Furthermore, to enforce the material balances, the models of the reaction and distillation tasks must relate the input and outlet flows using linear equations. The following sections derive models that overestimate the composition space that is reachable using batch distillation and batch reaction tasks.

3.1.2 Batch Distillation Composition Bounds

The targeting model of the batch distillation tasks, coupled with the opportunities for mixing embedded in the superstructure, must include all of the feasible sequences of cuts that could be obtained by any batch distillation column processing the same feed. Although we recognize that separating the mixture into its pure components represents a bound, the presence of azeotropes results in boundaries in the composition space that cannot usually be crossed. As a result, the sequence of products attained from batch distillation depends on the feed composition of the mixture. The location of these boundaries is likely to affect the solvents and entrainers chosen, the amount of solvent and reagent that is used, and the operation of the reactors providing the feeds to the distillation columns. Therefore, the targeting model must embed these boundaries in order to generate useful information during process development.

We model the distillation tasks shown in the superstructure using batch distillation targeting techniques (Ahmad and Barton, 1994; Ahmad and Barton, 1995) to identify the set of sharp splits that can be obtained from a given feed; we assume that sharp splits are possible when operating under the limiting conditions and the pot composition boundaries are linear (Ahmad and Barton, 1996). We then prove that the proposed superstructure contains all feasible sequences of cuts that can be achieved from a given feed, including non-sharp splits and off-cuts, in spite of the fact that we have represented distillation tasks shown in the superstructure using sharp splits.

Targeting for Sharp Splits

Simple residue curves describe the change in composition with time of an open evaporation process. These residue curves can be placed in the composition simplex defined by the pure component vectors to form a simple distillation residue curve map; an example map for a ternary system is shown in figure 3-2. These curves can be defined experimentally, or via the solution of a set of differential equations. Doherty and Perkins (1978a; 1978b; 1979) showed that the pure components and azeotropes represent the fixed points of a system of differential equations; further, all of the homogeneous azeotropes of a given system of components can be found using established algorithms (Fidkowski *et al.*, 1993). We let the fixed points arranged in increasing boiling temperature define the ordered set $E = \{\boldsymbol{\rho}_1, \boldsymbol{\rho}_2, \boldsymbol{\rho}_3, \ldots \boldsymbol{\rho}_{ep}\}$; ep represents the number of fixed points in the system, and $\boldsymbol{\rho}_e$ represents the composition of each fixed point.

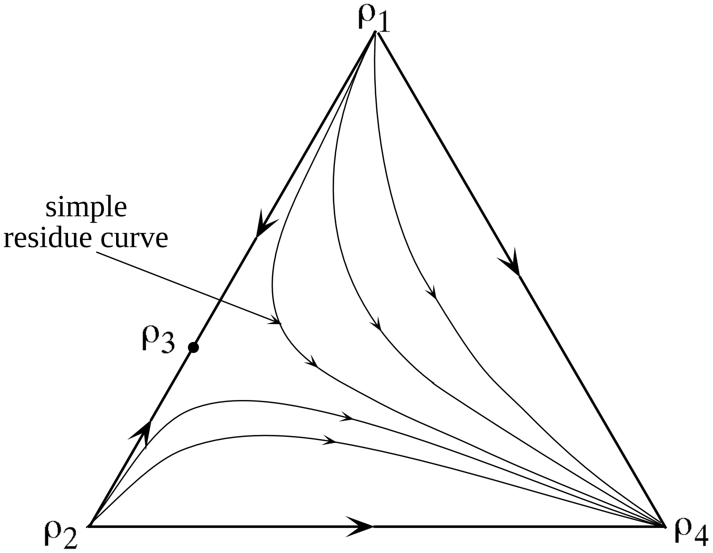

Figure 3-2: Residue curve map for a ternary system with pure components $\boldsymbol{\rho}_1$, $\boldsymbol{\rho}_2$, and $\boldsymbol{\rho}_4$. The fixed point $\boldsymbol{\rho}_3$ represents a maximum boiling binary azeotrope between $\boldsymbol{\rho}_1$ and $\boldsymbol{\rho}_2$.

Van Dongen and Doherty (1985) compared the simple distillation residue curves to the pot composition trajectory of a batch rectifier and demonstrated that the rectifying curves approach straight lines in the limit of high reflux ratio and a large number of equilibrium stages. Given a homogeneous ternary mixture under these limiting conditions, they showed that the exact orbit of the reboiler composition and the sequence of constant-boiling product cuts can be predicted from the structure of the residue curve map of the system. Under these limiting conditions, the composition simplex can be divided into a set of batch distillation regions. Each batch distillation region defines the set of compositions leading to the same sequence of product cuts. Figure 3-3 shows the batch distillation regions and trajectory of the reboiler composition for the residue curve map show in figure 3-2.

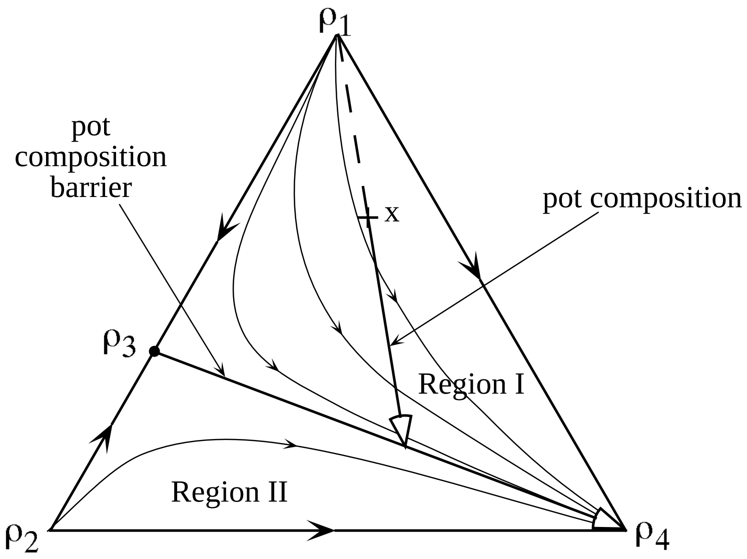

Figure 3-3: Ternary system with two distillation regions showing the pot composition trajectory for a feed in distillation region I.

Ahmad and Barton (1994; 1997) have extended and generalized these results to homogeneous systems with an arbitrary number of components. They demonstrated that under the assumptions of high reflux ratio, a large number of stages, and linear pot composition boundaries, a mixture of nc components will separate into at most nc product cuts. Therefore, each batch distillation region b is represented by an ordered subset of the fixed points, E_b, of dimension nc. These batch regions cover the nc component composition simplex.

$$\bigcup_{b \in B} b = \mathbf{C}^{nc} = \{\mathbf{x} \in \mathbb{R}^{nc} : \|\mathbf{x}\|_1 = 1, \;\; x_i \geq 0 \;\; \forall i = 1 \ldots nc\} \tag{3.1}$$

Furthermore, the members of E_b bound an $nc - 1$ dimensional simplex, termed the product simplex. The product simplex $P(b)$ is defined by an $nc \times nc$ matrix $\mathbf{P}^b$ as follows:

$$P(b) = \left\{ \mathbf{x} \in \mathbf{C}^{nc} : \mathbf{x} = \mathbf{P}^b \boldsymbol{\pi} \quad \forall \;\; \boldsymbol{\pi} \in \mathbf{C}^{nc} \right\} \tag{3.2}$$

where the columns of $\mathbf{P}^b$ correspond to the equilibrium point compositions appearing in the set E_b. Equation (3.2) defines the barycentric coordinates $\boldsymbol{\pi}$ representing the fraction of the charge appearing in each of the product cuts. Every batch region b defines a corresponding product simplex $P(b)$, but the converse is not always true (Ahmad and Barton, 1995). The targeting formulation presented here assumes that all batch regions coincide with their corresponding product simplices, so $P(b) = b$. For a given mixture of components, these regions can be determined from the stability of the fixed points (Ahmad *et al.*, 1997).

Given the product sequence defining each batch distillation region E_b and the compositions of all of the fixed points $\boldsymbol{\rho}_e$, we only need to identify the batch distillation region that contains the feed in order to perform the mass balance. We call the region containing the feed the active batch distillation region and identify it with the binary variable y^B. Since the feed lies within the convex hull of the products of the active region, the barycentric coordinates are positive. For regions that do not contain the feed, at least one of the barycentric coordinates is negative. We permit only one region to be active and require that the barycentric coordinates are positive ($\pi_{ke} \geq 0$), so we can express the fact that the feed composition $\mathbf{x}$ lies within the active region for the distillation task in train k as follows:

$$\sum_{b \in B} y^B_{kb} = 1 \qquad \forall \; k \in K \tag{3.3}$$

$$\mathbf{x}_k = \sum_{b \in B} y^B_{kb} \sum_{e \in E_b} \pi_{ke} \boldsymbol{\rho}_e \qquad \forall \; k \in K \tag{3.4}$$

We derive the time averaged mass balance for the distillation task by multiplying (3.4) by the total feed f^D. We define the variable $\mathbf{f}^{B_{out}}_b = y^B_b f^D \boldsymbol{\pi}$ to eliminate the bilinear terms from the time-averaged material balance and obtain the following material balance for the kth distillation task:

$$\mathbf{f}_k^{M_{out}} = \sum_{b \in B} \sum_{e \in E_b} f_{kbe}^{B_{out}} \boldsymbol{\rho}_e \qquad \forall\, k \in K \tag{3.5}$$

We require that $f_{kbe}^{B_{out}} \geq 0$ and complete the definition of $f_{kbe}^{B_{out}}$ using the following inequality:

$$\sum_{e \in E_b} f_{kbe}^{B_{out}} \leq f^{\max} y_{kb}^{B} \qquad \forall\, k \in K, b \in B \tag{3.6}$$

To simplify the expressions in the rest of the model, we define $f_{ke}^{D_{out}}$, the flow of equilibrium point e out of the distillation task k. Although this constraint is redundant, it will be eliminated during the preprocessing stage of the model (IBM, 1991) and will not effect the solver's efficiency.

$$\sum_{b \in B} \mathbf{f}_{kb}^{B_{out}} = \mathbf{f}_k^{D_{out}} \qquad \forall\, k \in K \tag{3.7}$$

The distillation targeting model presented above determines the maximum recovery for sharp splits. Now we prove that the superstructure embeds all feasible sequences of cuts that can be obtained from the same feed. Fractions of the sharp cuts can be combined to produce any feasible combination of cuts, embedding nonsharp splits and off cuts within the superstructure; therefore, the number of distillation cuts in the actual process need not correspond to the number of cuts in the targeting model as demonstrated in figure 3-4. A set of n cut compositions $S' = \{\boldsymbol{\rho}'_1, \boldsymbol{\rho}'_2, \ldots \boldsymbol{\rho}'_n : \boldsymbol{\rho}'_j \in \mathbf{C}^{nc} \quad \forall j = 1 \ldots n\}$ is feasible if and only if each cut is in the active batch distillation region ($\boldsymbol{\rho}'_j \in B^*$), and the feed composition $\mathbf{x}$ lies within the convex hull of the compositions in S' ($\mathbf{x} \in \mathrm{conv}(S')$). This definition does not imply that these compositions can actually be achieved in a column operating with a finite reflux ratio. Thus, the screening model embeds any off cuts and nonsharp splits that may be performed in the actual process.

Theorem 3.1. *Given a feed composition located in a batch distillation region B with* *linear pot composition boundaries that is identified by the sequence of product compositions $S = \{\boldsymbol{\rho}_1, \boldsymbol{\rho}_2, \ldots \boldsymbol{\rho}_{nc}\}$, all sets of feasible cuts can be obtained by mixing fractions of the cuts obtained from a column whose cut compositions are defined by S.*

Proof. Define the matrix $\mathbf{P} \in \mathbb{R}^{nc \times nc}$ as the matrix whose columns are the vectors in S and the matrix $\mathbf{P}' \in \mathbb{R}^{nc \times n}$ as the matrix whose columns are the vectors in S'. Since the batch distillation region is contained in the product simplex, each element of S' can be expressed as a convex combination of the elements of S, so there exists $\hat{\boldsymbol{\pi}}_j \in \mathbf{C}^{nc}$ such that $\boldsymbol{\rho}'_j = \mathbf{P}\hat{\boldsymbol{\pi}}_j$ for every $\boldsymbol{\rho}'_j \in S'$. This defines the matrix $\hat{\boldsymbol{\Pi}}$.

$$\mathbf{P}' = \mathbf{P}\hat{\boldsymbol{\Pi}} \tag{3.8}$$

Since $\mathbf{x} \in \text{conv}(S')$, there exists $\boldsymbol{\pi}' \in \mathbf{C}^{n}$ such that $\mathbf{x} = \mathbf{P}'\boldsymbol{\pi}'$ where π'_j represents the fraction of the charge obtained in the jth product cut of S'.

$$\mathbf{x} = \mathbf{P}'\boldsymbol{\pi}' = (\mathbf{P}\hat{\boldsymbol{\Pi}})\boldsymbol{\pi}' = \mathbf{P}(\hat{\boldsymbol{\Pi}}\boldsymbol{\pi}') \tag{3.9}$$

There exists $\boldsymbol{\pi} \in \mathbf{C}^{nc}$ defining the barycentric coordinates of the feed with respect to the extreme points of the distillation region, $\mathbf{x} = \mathbf{P}\boldsymbol{\pi}$, so the amounts collected in the sharp cuts are linearly related to any feasible cuts obtained from the column.

$$\hat{\boldsymbol{\Pi}}\boldsymbol{\pi}' = \boldsymbol{\pi} \tag{3.10}$$

This equation represents the material balance around the product cuts in the set S. It demonstrates that the amount of the cuts with the compositions in S' can be obtained by mixing fractions of the cuts taken at the equilibrium nodes. Figure 3-4 shows that any feasible set of cuts can be obtained from the sharp cuts determined in the targeting model if mixing is permitted. The labels on the arcs represent the time-averaged flow rates, and the labels contained in the state nodes denote the material composition. Since every element of both $\hat{\boldsymbol{\Pi}}$ and $\boldsymbol{\pi}'$ is positive, all of the flows on the arcs between the nodes are positive. □

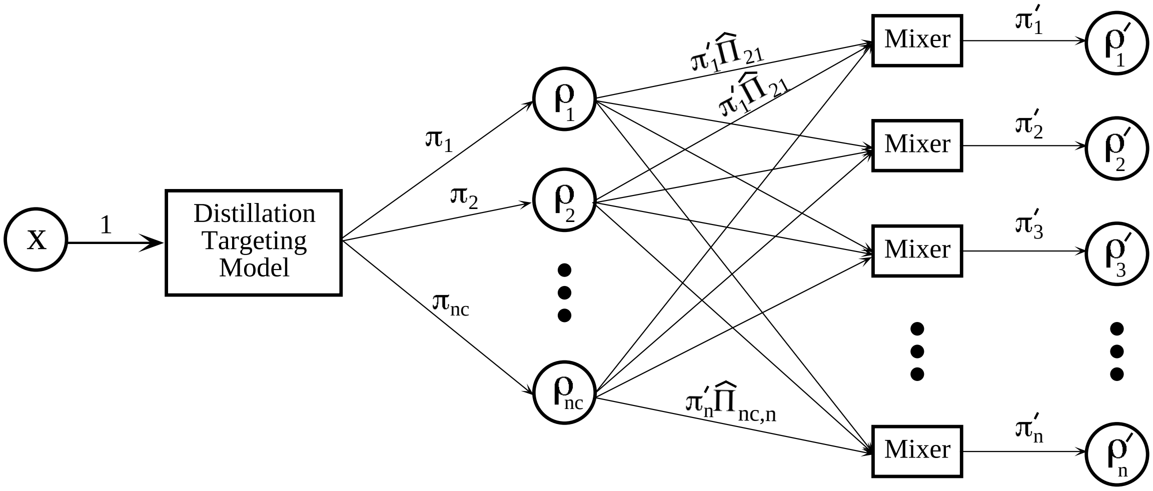

Figure 3-4: Representation of an arbitrary distillation task by combining sharp distillation cuts and mixers.

3.1.3 Reactor Targeting Model

Mass balances and reaction stoichiometry are enforced by introducing the extents of reaction as model variables. For the kth reaction task, stoichiometry is enforced by expressing the time averaged material balance in terms of stoichiometric coefficients $\boldsymbol{\nu}_{kr}$ and the extent ξ_{kr} of reaction r.

$$\sum_{e \in E} f_{ke}^{R_{in}} \boldsymbol{\rho}_e + \sum_{r \in R_k} \xi_{kr} \boldsymbol{\nu}_{kr} = \mathbf{f}^{R_{out}} \qquad \forall\, k \in K \tag{3.11}$$

For components e that do not participate in reaction r of the kth reaction task, $\nu_{kre} = 0$. Since the extent of the reaction is the same for all components, requiring non-negative flow rates insures that the reaction extents are feasible. The material balances for the reaction only constrain the feasible composition space of the reactions by enforcing stoichiometry and permitting no more than total conversion of any reactant.

The extents of the reaction that are achieved in the actual process depend on the operating policies of the reaction tasks and the kinetics of the reactions. Since expressions for the reaction kinetics are available (otherwise we could not model the reaction tasks in detail), bounds on the achievable extents of reaction in terms of key operating variables (e.g., processing time, temperature, and feed composition) can be derived and incorporated within the screening model. In addition, bounds relating the extents of competing reactions can be provided. We have not derived general expressions for these bounds since they will almost certainly depend on the kinetics of the reactions, but the case studies presented in chapters 4 and 5 show specific examples of how these bounds can be derived. The case studies demonstrate how bounds for the extents of competing reactions can be derived from the operating temperature limits imposed on the process. In addition, they demonstrate how upper bounds on the extents can be derived from the processing time and a bound on the temperature profile for the task. These bounds do not exclude any feasible operating policies, yet they manage to incorporate important tradeoffs within the screening formulation.

3.2 Time Averaged Material Balances

The constraints for the material balances can be derived from the superstructure, shown in figure 3-1, and the composition targeting models that relate the inlet and outlet flow rates for the distillation and reaction task nodes in the superstructure. In fact, the material balances for the distillation and reaction tasks are shown in (3.5–3.7) and (3.11) respectively. The screening model enforces time averaged material balances around each of the task and state nodes in the superstructure. Material balances around the state nodes representing the fixed points of the batch distillation regions are expressed as follows:

$$f_e^{Supply} + \sum_{k \in K} f_{ke}^{D_{out}} = f_e^{Product} + f_e^{Waste} + \sum_{k \in K} f_{ke}^{R_{in}} + \sum_{k \in K} f_{ke}^{M_{in}} \quad \forall\, e \in E \tag{3.12}$$

The following material balances around the 'hypothetical' mixing tasks define the feed to the distillation tasks in terms of pure component flows.

$$\sum_{e \in E} f_{ke}^{M_{in}} \boldsymbol{\rho}_e + \mathbf{f}_k^{R_{out}} = \mathbf{f}_k^{M_{out}} \quad \forall\, k \in K \tag{3.13}$$

Equations (3.5–3.7), (3.11), and (3.12–3.13) enforce the material balances around all of the nodes in the superstructure shown in figure 3-1; these constraints denote the material balance constraints at the highest level of the superstructure hierarchy. However, we cannot identify streams that are recycled and need to be purged by examining the superstructure at this level of detail. Since the screening models require that a fraction of any recycled cut is purged, deriving the purge constraints requires a more detailed view of the material flows in the process. The fixed point nodes in the superstructure shown in figure 3-1 are refined as shown in figure 3-5 to provide a superstructure with more detail that identifies recycled streams and allows them to be purged. Constraints to enforce the purge requirements require variables introduced in the material balance constraints for the network depicted in figure 3-5. In general, a hierarchy of superstructures may be used to describe the process, depending on the type of constraints that are required.

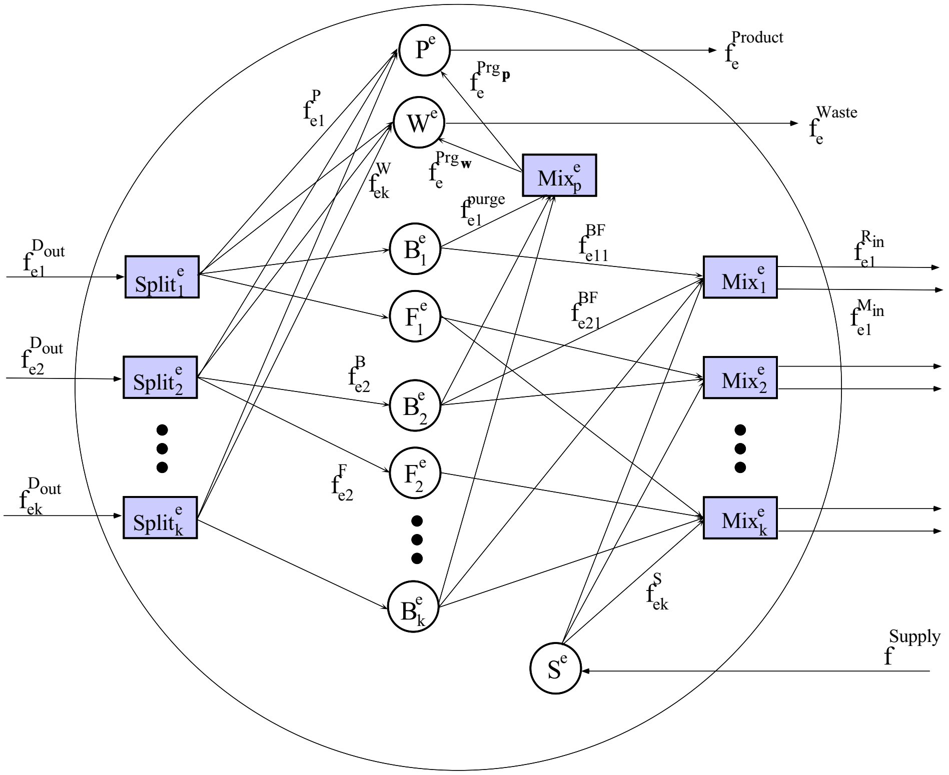

Figure 3-5: Detailed representation of fixed point node e used to derive the purge constraints.

The cuts from each distillation task are sent to a splitter contained in the detailed representation of the fixed point node. Cuts entering the network are either sent to waste, to product, forward in the process, or backward in the process. Material balances are derived around each node that exists in the expanded representation of the fixed point node in (3.14–3.22). Equation (3.22) ensures that a fraction of every recycled stream is purged. The purge fraction of each equilibrium point, X_e^{purge}, is data supplied to the screening model based on engineering judgment or prior knowledge about trace contaminants; different purge fractions can be used for each fixed point node if desired. Incorporating these constraints in the model, allows (3.12) to be removed from the screening model. We retain (3.14–3.22) and rely on the presolver contained in OSL to eliminate any unnecessary variables and constraints to reduce the size of the linear programs actually solved during the branch and bound iteration (IBM, 1991). If a solver is used that does not eliminate the intermediate variables that have been introduced here, these should be removed to reduce the size of the models that are solved.

$$f_{ke}^{D_{out}} = f_{ke}^{W} + f_{ke}^{Prod} + f_{ke}^{B} + f_{ke}^{F} \qquad \forall\, k, e \tag{3.14}$$

$$f_{ke}^{B} = f_{ke}^{purge} + \sum_{k' \leq k} f_{ekk'}^{BF} \qquad \forall\, k, e \tag{3.15}$$

$$f_{ke}^{F} = \sum_{k' > k} f_{ekk'}^{BF} \qquad \forall\, k, e \tag{3.16}$$

$$f_{ke}^{R_{in}} + f_{ke}^{M_{in}} = f_{ke}^{S} + \sum_{k'} f_{ek'k}^{BF} \qquad \forall\, k, e \tag{3.17}$$

$$f_{e}^{Supply} = \sum_{k} f_{ke}^{S} \qquad \forall\, e \tag{3.18}$$

$$f_{e}^{Product} = f_{e}^{Prg_p} + \sum_{k} f_{ke}^{P} \qquad \forall\, e \tag{3.19}$$

$$f_{e}^{Prg_w} + f_{e}^{Prg_P} = \sum_{k} f_{ke}^{purge} \qquad \forall\, e \tag{3.20}$$

$$\sum_{k} f_{ke}^{W} + f_{e}^{Prg_w} = f_{e}^{Waste} \qquad \forall\, e \tag{3.21}$$

$$f_{ke}^{purge} = X_{e}^{Purge} f_{ke}^{B} \qquad \forall\, k, e \tag{3.22}$$

The supply of raw material to the process is restricted to components that can be purchased or are available as a by product of another process within the manufacturing facility. Let E_R define the set of fixed points that may be supplied to the process and require that the feed of all other components is zero.

$$\sum_{e \notin E_R} f_e^{Supply} = 0 \tag{3.23}$$

Finally, the product must adhere to purity specifications and meet manufacturing demands. The total production is given by the flow of in-spec product over the entire campaign. Purity specifications are placed on a subset of the fixed points contained in the product (typically these will be pure components). We let E_P denote the components whose purity in the product is specified by $X^{product}$, and Q^{demand} represent the manufacturing demand. For example, if the desired product is component P and it is required at 98 % purity by mass, then the set $E_P = \{P\}$ and $X^{product} = .98$. The demand and purity constraints for the manufacturing campaign are specified below; in these constraints, the time averaged flow rates denote the material flow for the entire campaign, and the product purity is specified on a mass basis.

$$\begin{aligned} Q^{demand} &\leq \sum_{e \in E} f_e^{Product} \boldsymbol{\rho}_e^T \mathbf{w} && (3.24) \\ X^{product} \sum_{e \in E} f_e^{Product} w_e &\leq \sum_{e \in E} \left(f_e^{Product} \sum_{e' \in E_P} \boldsymbol{\rho}_e^T \boldsymbol{\rho}_{e'} w_{e'} \right) && (3.25) \end{aligned}$$

The elements of $\mathbf{w} \in \mathbb{R}^{nc}$ represent the molecular weights of the pure components. We could also place restrictions on the amounts of particular impurities that are permitted in the product. For example, if the product is required at 98 % purity, but cannot contain water, then a restriction must be placed on the amount of water that is allowed. Let the parameter $X_e^{impurity}$ denote the maximum mass fraction of fixed point e that is permitted in the product. If no special restrictions are imposed, then $X_e^{impurity} = 1 - X^{product}$ for all $e \notin E_P$, and $X_e^{impurity} = 1$ for all $e \in E_P$. Let the set E_I define the components whose concentration in the product is restricted to remain below the limit defined by $X_e^{impurity}$. Note that this set need only contain fixed points whose fraction in the product must be restricted more than the average impurity, such as water in the example described above.

$$X_e^{impurity} \sum_{e' \in E} f_{e'}^{Product} w_{e'} \geq \sum_{e' \in E} f_{e'}^{Product} \boldsymbol{\rho}_e^T \boldsymbol{\rho}_{e'} w_e \qquad \forall e \in E_I \tag{3.26}$$

Screening formulations containing objective functions that depend on only the material flows in the process can be derived using the constraints presented thus far (however, constraints that limit the extents of the reactions that were not explicitly stated should also be included). For instance, the minimum raw material and waste disposal cost for a process that meets the production requirements or the minimum amount of waste that can be emitted to the environment can be determined. We merely need to postulate the objective function, incorporate constraints (3.5–3.7, 3.11, 3.13, 3.14–3.23, and 3.24–3.26), and solve the resulting mixed-integer linear program. Similar models have been used for solvent recovery targeting (Ahmad and Barton, 1995). However, to account for other production costs and the assignment and scheduling of equipment, we need to target for the time and utility requirements for the reaction and distillation tasks and include constraints to account for the equipment assignment and scheduling. Such constraints are derived in the following sections.

3.3 Bounding Distillation Processing Time and Utility Requirements

The processing time and hot and cold utility consumption of the distillation task impact the operating cost of the entire batch process. Since the operating cost of the process is a nondecreasing function of these variables, underestimates are required to maintain the bounding properties of the screening model. However, determining the processing time and utility cost requires knowledge of both the reflux ratio and the amount of material taken overhead. This requires knowledge of the amount of material assigned to the bottoms, f_{ke}^{Bot}, defined later in this section.

3.3.1 Distillation Processing Time Bounds

The distillation columns employed in the process are characterized by a maximum vapor rate at which they can operate. The maximum vapor rate is based on limits imposed by the tray and downcomer design (or packing design) that avoids entrainment flooding for reasonable values of the liquid rate in the column (Kister, 1992). We assume that no loss of efficiency or increase in utility cost is incurred by operating at this rate. We also assume that no heat integration will be performed. Since operating at the maximum vapor rate will minimize the operating time but will not hinder separation efficiency or increase utility cost, all columns will operate at their maximum vapor rate.

The material balance around the column is used to derive bounds on the processing time and utility requirements. The column contains product cuts c to nc at the start of the cth product cut; at the completion of the cut, cuts $c+1$ to nc remain. The amount and composition of the material removed is known,[1] so the processing time can be calculated from the vapor and distillate rates. We assume that the vapor flow rate V is bounded by the maximum rate that can be achieved in a given column; no assumptions are made regarding the distillate rate D, or the liquid rate L.

To preserve the bounding property of the screening model, a valid underestimate of the operating time is needed. The time required to obtain each cut depends on the amount of the cut, the vapor rate, and the reflux ratio used during the cut. To provide a lower bound, we assume that the columns assigned to the distillation task will operate at their maximum vapor rates. Although the amount of material obtained in each product cut is given by $f_e^{D_{out}}$, when more than one unit is assigned, the amount of material processed by each column will be a fraction of $f_e^{D_{out}}$. In the remainder of this section, we consider $f_e^{D_{out}}$ to represent the material processed by the

[1] None of the material assigned to the bottoms cuts is taken overhead, providing an underestimate of the time and utility requirements. However, some of the overhead cut material may leave the column as an impurity in the bottoms stream, and this is addressed later in this section.

assigned equipment units; we adjust for units in parallel (see (3.66)) when deriving the constraints to determine the campaign time.

The processing time for each cut, t_e^{cut}, is the time required to remove the cut from the column. This time is a function of the distillate rate D and can be expressed in terms of the vapor rate and reflux ratio R. Let M represent the amount of material collected in the accumulator during the cut ($dM = Ddt$) and integrate the expression $V = D(1+R)$ for the duration of the product cut.[2]

$$\int_0^{t_e^{cut}} V dt = V t_e^{cut} = \int_0^{f_e^{D_{out}}} (1 + R(M))dM \tag{3.27}$$

The relationship above holds as long as the reflux policy can be expressed as a function of the amount collected in the accumulator during a specific cut. If the reflux ratio is constant over the entire cut, a simple expression for the time is obtained from (3.27).

$$t_e^{cut} = \frac{f_e^{D_{out}}(1+R)}{V} \tag{3.28}$$

The cut time defined in (3.28) provides a valid underestimate of the processing time for a cut if R underestimates the integral of the reflux ratio over the entire cut, $R \leq \int_0^{f_e^{D_{out}}} R(M)dM/f_e^{D_{out}}$.

In order to obtain an underestimate of the reflux ratio, some limiting cases are examined. First, since the column is operating at its maximum vapor flow rate, we recognize that a minimum reflux ratio is required to provide a suitable liquid rate for proper liquid and gas flow patterns within the column. This minimum ratio may depend on the particular column, and is required to prevent undesirable operating phenomena. Kister (1990; 1992) describes correlations to predict these boundaries for tray and packed columns, so we treat these boundaries as design constraints that cannot be violated. Thus, we assume that a minimum reflux ratio for the column is specified as part of the data for the problem. At the very least, any feasible operating

[2] Note that this relationship does not assert constant molar overflow. The vapor rate V is the maximum vapor rate that can be achieved in any part of the column. The vapor rate at the top stage must be less than or equal to V, so the distillate rate D must be less than or equal to $V/(1+R)$.

policy must employ a reflux ratio that exceeds this minimum. Since the equilibrium stage models will not accurately represent the process if we operate below this minimum, we should also include this constraint in any dynamic optimization calculations performed on the detailed models of the distillation tasks. If no information regarding the purity of the overhead cuts is provided, then the tightest bounds that can be given for the reflux ratio are those at the limit of the feasible operating regime based on liquid gas contacting. Letting $R_i^{\min}$ represent the minimum reflux ratio of the assigned equipment unit,

$$R_i^{\min} \leq R \leq \frac{\int_0^{f^{D_{out}}} R(M)dM}{f^{D_{out}}} \tag{3.29}$$

An underestimate of the processing time for the distillation task is obtained by adding the processing time for all of the overhead cuts, provided that the bottoms stream is pure. If the bottoms stream contains some impurities from the overhead stream, then some of the material that would have been taken overhead remains in the bottoms. To account for the impurity when determining the duties for the overhead cuts, we require that the amount of impurity that can be tolerated in the bottoms, $1 - X_k^{BP}$, is specified for each distillation task. The bottoms impurities must be fractions of the overhead cuts, so they can be defined as follows:

$$f_{ke}^{BI} \geq f_{ke}^{Bot} \quad \forall k,\ e \tag{3.30}$$

$$\sum_e f_{ke}^{BI} \leq \left(1 - X_k^{BP}\right) \sum_e f_{ke}^{Bot} \quad \forall k \tag{3.31}$$

Valid bounds are obtained by subtracting the time required to collect the tolerated amount of impurity at the reflux ratio employed during the overhead cut; the optimization is free to select the overhead material that minimizes the processing time as the impurity. Therefore, operating column i at its minimum reflux ratio defines the minimum time for one column of type i to distill the material taken overhead in distillation task k.

$$t_{ki}^{D_{proc}} \geq \frac{1 + R_i^{\min}}{V_i} \sum_e f_{ke}^{D_{out}} - f_{ke}^{Bot} - f_{ke}^{BI} \tag{3.32}$$

Of course, $f_{ke}^{BI} = 0$ and (3.30–3.31) are not needed if the bottoms streams are required to be pure.

3.3.2 Bounding the Distillation Utility Requirements

The rate of energy removal, $\dot{Q}$, required to condense the vapor passing through the condenser for a process operating without losses can be expressed in terms of the heat of vaporization of the condensate ΔH^{vap} and the reflux ratio R of the cut.

$$\dot{Q} = \Delta H^{\text{vap}} D(1 + R) \tag{3.33}$$

The distillate composition corresponds to one of the equilibrium points in the residue curve map, so ΔH^{vap} is known for every cut if the material is condensed at its boiling temperature; the enthalpy of vaporization and boiling temperature of each equilibrium point can be provided as data to the screening model.[3] However, we cannot assume that all of the material that is collected overhead is condensed at the boiling point of the fixed point because the cuts that will actually be obtained in the real column cannot achieve the limit of perfect splits. When the cuts are not sharp, a particular fixed point will be condensed as part of a mixture, so some fixed points will be condensed at a temperature above their normal boiling point. At these elevated temperatures, the enthalpy of vaporization is less than that at the normal boiling point because the enthalpy of vaporization is a decreasing function of temperature

[3] The enthalpy of vaporization must be underestimated for the fixed points. These underestimates must account for the enthalpy of mixing at the boiling temperature. The maximum enthalpy of mixing can be determined by formulating and solving a global optimization problem. The global optimization is solved before the screening model is posed, and the solution is treated as data in the screening model, so ΔH_e^{vap} represents the enthalpy of vaporization at the boiling point reduced by ΔH^{mix}. In principle, global optimization techniques (Adjiman *et al.*, 1996; Maranas and Floudas, 1996; Smith and Pantelides, 1995) can be employed to identify ΔH^{mix} for the compositions and temperatures considered using the enthalpy model employed during the detailed dynamic simulation.

(Reid *et al.*, 1987). This implies that a lower bound on the condenser duty is not derived by simply assuming that the collected material is condensed at its boiling temperature and the column operates at minimum reflux. However, the enthalpy of vaporization at the boiling temperature can be used to bound the reboiler duty.

We assume that material charged to the column is a liquid mixture below the boiling temperature of the fixed points collected in the overhead cuts. In order to collect material overhead, vapor must be generated. We adjust for the changes of enthalpy upon mixing separately when underestimating the energy requirements, so we ignore mixing effects here and treat the mixture as if it is ideal. Let ΔH_e denote the difference between the molar enthalpy of the liquid of fixed point e charged to the column and the molar enthalpy of the vapor generated in the reboiler at some point during the operation of the column. For a column operating at constant pressure, a lower bound on the energy supplied to the reboiler during the distillation can be determined from the amount of material taken overhead, the heat of vaporization of this material, and the reflux policy employed:

$$\sum_{e \in \mathrm{Ovhd}} \hat{Q}_{ke} = \sum_{e \in \mathrm{Ovhd}} \int_0^{f_{ke}^{D_{out}}} \Delta H_e (1 + R(M_e)) dM_e \tag{3.34}$$

where M_e represents the amount of material collected during cut e. A rigorous underestimate of the reboiler duty is obtained from (3.34) when a valid underestimate of the integral is provided; this requires valid underestimates for ΔH_e and the reflux ratio as functions of M_e and the temperature of the reboiler. A simple underestimate of the reflux ratio is obtained by assuming that the column operates at the minimum reflux $R_i^{\min}$ during the entire cut. Next, we demonstrate that the enthalpy of vaporization at the boiling temperature of the fixed points ($\Delta H_e^{\mathrm{vap}}$) provides a valid underestimate of ΔH_e.

The enthalpy of vaporization at the boiling temperature of fixed point e underestimates the difference in enthalpy between the liquid of fixed point e charged to the column and the vapor that is generated in the reboiler. To prove this statement we consider two cases: vapor that is generated below the boiling temperature, and vapor that is generated above the boiling temperature. The distillation is assumed to be carried at constant pressure, so we are concerned with the enthalpy change in an isobaric process. Let T_e^b represent the normal boiling temperature of fixed point e, T^{vap} represent an arbitrary temperature at which vapor is generated, T^{in} represent the temperature of the feed to the column, $\Delta H_e^v(T^{\text{vap}})$ represent the enthalpy of vaporization of fixed point e at T^{vap}, and ΔH_e^{vap} represent the enthalpy of vaporization at T_e^b.

First consider the case in which vapor is generated below the boiling temperature (e.g., $T^{\text{vap}} \leq T_e^b$). The enthalpy difference between the liquid charged and saturated vapor at T^{vap} can be expressed as follows:

$$\Delta H_e(T^{\text{vap}}) = \int_{T^{in}}^{T^{\text{vap}}} C_{p_e}^l(T)dT + \Delta H_e^v(T^{\text{vap}}) \tag{3.35}$$

Since the enthalpy of vaporization is a decreasing function of temperature, $\Delta H_e^{\text{vap}} < \Delta H_e^v(T^{\text{vap}})$. In addition, C_{p_e} is positive, and we assume $T^{in} < T^{\text{vap}}$, so substituing into (3.35) provides an underestimate of the enthalpy change required to generate vapor of fixed point e below the boiling temperature.

$$\Delta H_e(T^{\text{vap}}) \geq \Delta H_e^{\text{vap}} \tag{3.36}$$

On the other hand, if the vapor is generated at or above the boiling temperature (e.g., $T^{\text{vap}} \geq T_e^b$) then the enthalpy difference between the liquid charged and the vapor obtained can be described by the following isobaric path:

$$\Delta H_e(T^{\text{vap}}) = \int_{T^{in}}^{T_e^b} C_{p_e}^l(T)dT + \Delta H_e^v(T_e^b) + \int_{T_e^b}^{T^{\text{vap}}} C_{p_e}^v(T)dT \tag{3.37}$$

Since the temperatures are ordered ($T^{in} \leq T_e^b \leq T^{\text{vap}}$) and the vapor and liquid heat capacities are positive, $\Delta H_e(T^{\text{vap}})$ is also underestimated by ΔH_e^{vap} when the vapor is generated at temperatures above T_e^b.

$$\Delta H_e(T^{\text{vap}}) \geq \Delta H_e^{\text{vap}} \tag{3.38}$$

Thus, the enthalpy of vaporization at the boiling temperature underestimates the enthalpy difference between vapor at temperatures greater than T_e^b and liquid at T^{in}. Therefore, an underestimate of the reboiler duty of distillation k can be expressed as follows:

$$\sum_{e \in \text{Ovhd}} \hat{Q}_{ke} \geq \sum_{e \in \text{Ovhd}} \Delta H_e^{\text{vap}} (1 + R_i^{\text{min}}) \quad \forall\, k \in K \tag{3.39}$$

We note that for an exothermic reactive distillation process this may not be the case, and the heat of reaction would need to be considered when determining the bound on the reboiler duty. However, we do not consider reactive distillation in this thesis.

The energy costs in this type of process are typically unimportant, so these crude underestimates of the utility requirements do not really influence the important design trade offs. As mentioned in chapter 1, the small energy requirements of these processes is one of the properties that favors their manufacture in developed nations. The example problems presented in chapters 4 and 5 demonstrate that the utility costs are insignificant in comparison to the other manufacturing costs. In fact, these costs would still be insignificant even if they were an order of magnitude greater.

An underestimate of the duty for the distillation task is obtained by adding the duties for all of the overhead cuts, provided that the bottoms stream is pure. Valid bounds are obtained by subtracting the duty required to collect the tolerated amount of impurity at the reflux ratio employed during the overhead cut; the optimization will select the overhead material with the greatest heat of vaporization as the impurity. Thus, for a column operating at vapor rate of V and a constant reflux ratio R satisfying (3.29), the minimum reboiler duty can be defined as follows:

$$Q_k = \left(1 + \sum_{i \in I_D} \sum_{n=1}^{N_i} y_{ikn}^R R_i^{\text{min}} \right) \sum_e \Delta H_e^{\text{vap}} \left(f_{ke}^{D_{out}} - f_{ke}^{Bot} - f_{ke}^{BI} \right) \quad \forall k \in K \tag{3.40}$$

3.3.3 Definition of Bottoms Cuts

Whether a separation task is performed or not is determined from the location of the bottoms cut in the distillation task. If all of the material entering the column is taken in the bottoms, then the distillation is not performed and the processing time and utility requirements defined above would both be zero. Therefore, every distillation task in the superstructure must define which fixed point in the cut sequence will be the first that is included in the bottoms; $y_{ke}^{Bot} = 1$ denotes that e is the first product taken in the bottoms of distillation k. We require a bottoms cut for every distillation task, so

$$\sum_{e \in E} y_{ke}^{Bot} = 1 \qquad \forall\, k \in K \tag{3.41}$$

and we require that the bottoms cut exists in the active batch distillation region

$$y_{ke}^{Bot} \leq \sum_{b \in B_e} y_{kb}^{B} \qquad \forall\, e \in E,\ k \in K \tag{3.42}$$

where B_e represents the set of all batch regions containing fixed point e (e.g., $B_e = \{b \in B : e \in E_b\}$). Any cut appearing after the bottoms cut in the product sequence will be taken in the bottoms as well, so the bottoms of the distillation task can be defined as follows:

$$f_{ke}^{Bot} = f_{ke}^{D_{out}} \sum_{e' \leq e} y_{ke'}^{Bot} \qquad \forall\, e \in E,\ k \in K \tag{3.43}$$

We require that all of the bottoms cuts are processed in the same fashion. The bottoms may be passed on to the next reaction or mixing task, or out of the process as product or waste. If the bottoms stream is comprised of only one fixed point (i.e., the last cut in the active batch distillation region), then it may be processed in the same way as any other cut. The constraints defining the way that the bottoms are processed are given below.

$$\sum_{s \in S} y^S_{ks} = 1 \qquad \forall k \in K \tag{3.44}$$

$$f^{R_{in}}_{k+1,e} \geq f^{Bot}_{ke} y^S_{k,\text{rxn}} \qquad \forall k, e \tag{3.45}$$

$$f^{M_{in}}_{k+1,e} \geq f^{Bot}_{ke} y^S_{k,\text{mix}} \qquad \forall k, e \tag{3.46}$$

$$f^{Prod}_{ke} \geq f^{Bot}_{ke} y^S_{k,\text{prod}} \qquad \forall k, e \tag{3.47}$$

$$f^{W}_{ke} \geq f^{Bot}_{ke} y^S_{k,\text{waste}} \qquad \forall k, e \tag{3.48}$$

The bottoms may only be sent anywhere if the cut is the last cut taken from the active batch distillation region denoted by e^b_{nc} (i.e., the ncth cut from the region).

$$y^S_{k,\text{any}} \leq \sum_b y^B_{kb} y^{Bot}_{ke^b_{nc}} \qquad \forall k \tag{3.49}$$

3.4 Equipment Allocation

The product will be manufactured in a single product campaign using a subset of the equipment available within the manufacturing facility. Suitable equipment items must be assigned to all of the tasks that are performed in the process. Processing tasks can employ parallel items of equipment, but only identical columns are permitted at the same processing stage. Allocation and overflow constraints are enforced, and the performance of the process is analyzed for two storage policies — no intermediate storage and unlimited intermediate storage.

Since a suitable item of equipment must be assigned to every task that is performed, we require variables to define whether a task is performed. Let y^{Rxn}_k and z^D_k define the existence of reaction and distillation task k, respectively. A distillation task is performed unless the first cut from the active batch distillation region is included in the bottoms. Letting e^b_1 denote the index of the first cut in region b, the existence of the kth distillation task is defined as follows:

$$z_k^D = 1 - \sum_{b \in B} y_{ke_1^b}^{Bot} y_{kb}^B \qquad \forall\, k \in K \tag{3.50}$$

If a reaction task is not performed then all the extents of reaction are zero.

$$\sum_{r \in R_k} \xi_{kr} \leq y_k^{Rxn} f^{\max} \qquad \forall k \tag{3.51}$$

The screening model permits material to flow into tasks that are not performed but the equipment overflow constraints are relaxed, so no equipment needs to be assigned. For the columns, (3.43) requires that all of the material leaves these tasks in the bottoms if the distillation is not performed. Equations (3.52–3.53) ensure that equipment is assigned to the reactions and distillations that are performed.

$$\sum_{i \in I_R} \sum_{n=1}^{N_i} z_{ikn}^R \geq y_k^{Rxn} \qquad \forall\, k \tag{3.52}$$

$$\sum_{i \in I_D} \sum_{n=1}^{N_i} y_{ikn}^C = z_k^D \qquad \forall\, k \tag{3.53}$$

The equipment items of type i assigned to the process cannot exceed the number of equipment items, N_i, of that type available in the plant's inventory.

$$\sum_{n=1}^{N_i} \sum_{k} y_{ikn}^R n \leq N_i \qquad \forall\, i \in I_R \tag{3.54}$$

$$\sum_{n=1}^{N_i} \sum_{k} y_{ikn}^C n \leq N_i \qquad \forall\, i \in I_D \tag{3.55}$$

We also require that parallel distillation columns are the same type.

$$\sum_{i \in I_D} \sum_{n=1}^{N_i} y_{ikn}^C \leq 1 \qquad \forall\, k \in K \tag{3.56}$$

Consecutive reaction tasks may be merged if the distillation task between them is not performed; if a distillation is not performed, the optimization is free to choose whether the adjacent reactions should be merged into the same equipment items. Let y_k^{merge} denote whether reaction k is merged with reaction $k+1$, z_{ikn}^R denote the whether n equipment items of type I are assigned to reaction task k, and y_{ikn}^R denote the first reaction task to which these equipment items are assigned.

$$y_k^{merge} \leq 1 - z_k^D \quad \forall\, k < K \tag{3.57}$$

If two consecutive reaction tasks are merged, then the same equipment items are used for each task. This implies that no new equipment items are assigned to the latter stage which is enforced by (3.58).

$$y_{k-1}^{merge} + \sum_{n=1}^{N_i} y_{ikn}^R \leq 1 \qquad \forall\, i \in I_R,\ k > 1 \tag{3.58}$$

Using the fact that no new equipment is assigned, the variable z_{ikn}^R can be defined recursively as follows:

$$z_{i,k-1,n}^R y_{k-1}^{merge} + y_{ikn}^R = z_{ikn}^R \qquad \forall i \in I_R,\ k \in K, n \tag{3.59}$$

where $z_{i,0,n}^R = 0$ and $y_0^{merge} = 0$. Equation (3.59) can be expressed using the following linear constraints since $z_{ikn}^R - y_{ikn}^R = z_{i,k-1,n}^R y_{k-1}^{merge}$:

$$z_{ikn}^R - y_{ikn}^R \leq z_{k-1,in}^R \qquad \forall\, k, i \in I_R, n \tag{3.60}$$

$$z_{ikn}^R - y_{ikn}^R \leq y_{k-1}^{merge} \qquad \forall\, k > 1, i \in I_R, n \tag{3.61}$$

$$z_{ikn}^R - y_{ikn}^R \geq z_{k-1,in}^R + y_{k-1}^{merge} - 1 \qquad \forall\, k, i \in I_R, n \tag{3.62}$$

$$z_{ikn}^R - y_{ikn}^R \geq 0 \qquad \forall\, k, i \in I_R, n \tag{3.63}$$

Note that equations (3.60–3.62) are the standard linearization proposed by Glover (1975) for bilinear terms of binary variables, but (3.63) is required to ensure that z_{ikn}^R is equal to y_{ikn}^R at the first stage to which equipment is assigned.

3.5 Process Performance and Production Cost

The equipment assigned to the processing tasks and the storage policy selected for the process affect the production rate of the process and the duration of the manufacturing campaign. Since the reaction times do not depend on the item of equipment that is used, and identical distillation columns are assigned to the same task, an unlimited intermediate storage policy (UIS) is modeled by treating the number of batches of each task as an integer variable. $N_k^{batch_R}$ and $N_k^{batch_D}$ represent the number of batches used for the reaction and distillation task in train k. The number of batches for tasks that are not performed is arbitrarily assigned to the maximum number of batches. The no intermediate storage policy (NIS) is modeled by requiring that the number of batches used for every task is the same, and the arbitrary assignment for unperformed tasks is relaxed. The model equations below are derived for the UIS case, recognizing that the NIS case can be derived by adding constraints, or substituting N^{batch} for both $N_k^{batch_R}$ and $N_k^{batch_D}$. Letting the time averaged flows represent the total flows over the duration of the campaign, the following constraint underestimates the processing volume required for the reactors and represent a relaxation of the constraint requiring that the reactors do not overflow:

$$\sum_e f_{ke}^{R_{out}} \boldsymbol{\rho}_e^T \mathbf{v} \leq \sum_{i \in I_R} \sum_{n=1}^{N_i} z_{ikn}^R n N_k^{batch_R} \hat{V}_i + N^{B^{\max}} V^{C_{\max}} \left(1 - y_k^{Rxn}\right) \qquad \forall\, k \qquad (3.64)$$

where $\mathbf{v}$ is a vector whose components underestimate the molar volume of each of the pure components in the process over the temperature range of interest. If volume changes upon mixing are modeled, these underestimates must be chosen so that valid underestimates are still obtained for the resulting mixture volumes when the volume is calculated as if it is an ideal mixture.[4] Note that the volume requirement is based solely on the underestimate of the final reaction volume in order to account for fed

[4] To account for volume changes, the molar volume of each component is adjusted to account for the maximum volume change upon mixing that is possible over the temperature and compositions considered. This maximum change can, in principle, be calculated by applying global optimization techniques (Adjiman *et al.*, 1996; Maranas and Floudas, 1996; Smith and Pantelides, 1995) to the mathematical model used to predict liquid volume in the detailed dynamic models.

batch operating policies, and that the constraint is relaxed if the reaction task is not performed and the contents are passed on to the subsequent distillation task. If the reactions must run in batch mode, then a similar constraint can be imposed on the initial reactor volume. Detailed simulation of a reactor with these feed flows may actually overflow since these constraints overestimate the feasible region. Similar constraints are enforced for the distillation columns, but we assume that all of the material is charged to the column at the start of the task.

$$\sum_e \mathbf{v}^T \mathbf{f}_e^{M_{out}} \leq \sum_{i \in I_D} \sum_{n=1}^{N_i} n y_{ikn}^C N_k^{batch_D} \hat{V}_i + N^{B^{\max}} V^{C_{\max}} \left(1 - z_k^D\right) \qquad \forall\, k \tag{3.65}$$

The campaign time for the process depends upon the processing times for the individual tasks. The processing time for each distillation task depends upon the columns assigned and the amount of material processed. Parallel distillation columns are required to be of the same type, so an optimum exists with equal amounts of material sent to each. Thus, the processing time for columns operating at the minimum allowable reflux ratio of $R_i^{\min}$ to complete distillation task k is given as follows:

$$t_k^D = \left(\sum_e f_{ke}^{D_{out}} - f_{ke}^{Bot} - f_{ke}^{BI} \right) \sum_{i \in I_D} \sum_{n \leq N_i} \frac{z_{ikn}^C (1 + R_i^{\min})}{n V_i} \tag{3.66}$$

The reaction processing times t_k^R for one batch are independent of the assigned equipment units, yet we need to consider whether the reaction tasks are merged to determine the total batch processing time for reactors assigned to these tasks.

$$t_k^{merged} = t_k^R + y_{k-1}^{merge} t_{k-1}^{merged} \quad \forall\, k \tag{3.67}$$

The total processing time needs to consider the transfer times and any time allotted to bring the columns to total reflux. Constant transfer times are assumed, leading to the following bounds on the campaign time.

$$t^{campaign} \quad \geq \quad t_k^D + N_k^{batch_D} \left(t^{charge} + t^{empty} + t^{reflux}\right) \quad \forall\, k \tag{3.68}$$

$$t^{campaign} \geq N_k^{batch_R}\left(t_k^{merged} + t^{charge} + t^{empty}\right) \quad \forall\, k \tag{3.69}$$

In addition, the time available for manufacture is typically restricted.

$$t^{campaign} \leq t^{horizon} \tag{3.70}$$

The cost of manufacture includes raw material, waste disposal, equipment use, and utility costs. Each equipment item has associated an hourly rental charge. Equipment items must be rented for the entire campaign, so the equipment cost for the campaign can be expressed as follows:

$$c^{equip} = t^{campaign}\sum_{i\in I_D}\sum_{n=1}^{N_i} n z_{ikn}^{C} C_i^E + t^{campaign}\sum_{i\in I_R}\sum_{n=1}^{N_i} n z_{ikn}^{R} C_i^E \tag{3.71}$$

Utility costs are calculated from the duties for distillation tasks and cost of the specific utility required. Below, we assume only one level of the hot and cold utility is available, although this is not necessary in general.

$$\left(C^{hu} + C^{cu}\right)\sum_k Q_k = c^{utility} \tag{3.72}$$

Raw material and waste disposal charges are associated with every fixed point node. Total waste and raw material costs are determined from the total mass of material entering and leaving the process.

$$c^{raw} = \sum_{e\in E_R} C_e^r f_e^{Supply} \tag{3.73}$$

$$c^{waste} = \sum_{e\in E} C_e^w f_e^{Waste} \tag{3.74}$$

An underestimate of the total manufacturing cost is given as the sum of the individual costs.

$$c^{total} = c^{raw} + c^{waste} + c^{utility} + c^{equip} \tag{3.75}$$

3.6 Formulating the model to be solved

The constraints presented above permit the minimization of the underestimate of the manufacturing cost expressed in (3.75) subject to constraints (3.5–3.7), (3.11), (3.13–3.25), (3.30–3.40), (3.41–3.58), and (3.60–3.74). However, the model, as presented, cannot be solved to guaranteed global optimality since it is nonconvex. All of the nonconvexities in the formulation arise from bilinear terms between discrete variables or between discrete and continuous variables; these terms are present in (3.40), (3.43–3.49), (3.50), (3.64–3.67), (3.69), and (3.71). Since exact linearizations of these expressions are possible, the model can be transformed into a mixed-integer linear program that can be solved to guaranteed global optimality (Glover, 1975; Adams and Sherali, 1986).

The bilinear products of two binary variables are modeled by defining continuous variables that are an exact linearization of the bilinear product. For example, the bilinear product $y^{Bot}_{ke^b_1} y^B_{kb}$ appearing in (3.50) is replaced by introducing the continuous variable $z^{B_1}_{kb}$ equal to the bilinear product that is defined in terms of linear constraints following the linearization scheme proposed by Glover (1975):

$$z^{B_1}_{kb} \leq y^{Bot}_{ke^b_1} \quad \forall\, b,k \tag{3.76}$$

$$z^{B_1}_{kb} \leq y^B_{kb} \quad \forall b,k \tag{3.77}$$

$$z^{B_1}_{kb} \geq y^{Bot}_{ke^b_1} + y^B_{kb} - 1 \quad \forall\, b,k \tag{3.78}$$

The bilinear terms of continuous and discrete variables are also linearized following the scheme proposed by Glover (1975) that exploits the upper and lower bounds (e.g., (3.81)) on the continuous variables. For example, the variable t^{RM}_k is introduced to replace the bilinear term in (3.67).

$$t^{merged}_k - t^{merged^+}_k (1 - y^{merge}_k) \leq t^{RM}_k \leq t^{merged}_k - t^{merged^-}_k (1 - y^N_n) \quad \forall\, k < K \tag{3.79}$$

$$t^{merged^-}_k y^{merge}_k \leq t^{RM}_k \leq t^{merged^+}_k y^{merge}_k \quad \forall\, k < K \tag{3.80}$$

$$t^{merged^-}_k \leq t^{RM}_k \leq t^{merged^+}_k \tag{3.81}$$

These constraints typically increase the integrality gap of the model. Finding tight upper and lower bounds on the variables helps to mitigate this effect; calculations to estimate tight bounds on the variables are discussed in chapter 4. Additional constraints can also be introduced to derive a tighter formulation (Adams and Sherali, 1986).

The integer variables representing the number of batches are modeled as the sum of binary variables to enable standard linearization techniques to be applied. Special ordered sets of type 1 are used for these binary variables to improve the efficiency of the solver's branch and bound iteration (Beale and Tomlin, 1970).

$$N^{batch} = \sum_{m=1}^{N^{B^{\max}}} m y_m^{NB} \tag{3.82}$$

$$\sum_{m=1}^{N^{B^{\max}}} y_m^{NB} = 1 \tag{3.83}$$

3.7 Conclusions

Screening models for batch process development have been derived. A superstructure for networks of batch reaction/distillation tasks has been presented. This superstructure embeds sequences of reaction and distillation tasks with material recycles. Equations to enforce time averaged material balances for the nodes in the superstructure have been derived. Composition targets for the reaction and distillation tasks overestimate the feasible region of operation and enforce mass balances for the tasks. Although the distillation targeting model assumes sharp splits, we have demonstrated that the superstructure embeds all feasible sequences of distillation cuts. In addition, the modeling equations for the reaction and distillation tasks provide rigorous underestimates of the processing time and utility requirements. The distillation targets that have been derived show that when the minimum reflux ratio is determined from the limit required for proper gas/liquid contacting, the screening model can be cast as a mixed-integer linear program. Within this formulation, the screening models address the allocation of equipment to processing tasks for both UIS and NIS storage policies, and consider raw material, waste disposal, utility, and equipment costs.

The screening models provide a rigorous lower bound on the cost of the design. This lower bound can be employed as a design target to enhance existing design methods, or as the basis for a rigorous decomposition algorithm to address batch process development. For instance, the solution of the screening model can be employed as a metric upon which the benefits of design optimization can be assessed, and it can be used to determine whether a new product has any chance of being profitable. Screening models also enable the development of the rigorous decomposition strategy for the improvement of the design, discussed in section 2.4, that has the potential to avoid total enumeration of the discrete space. The decomposition strategy also provides a rigorous bound the distance to the global solution upon termination.

In addition, the screening models consider aspects of the batch process synthesis that have not previously been systematically addressed. Solvents and reagents can be selected from a set of candidates and the models can determine the sequence of processing tasks from a superstructure of processing alternatives. The solution constructs not only the sequence of tasks to be performed, but also defines the recycle structure for the process. For these reasons, the solution provided by the screening model provides a good starting point for detailed design. This solution facilitates the definition of a state task network of the process that can be used to formulate the detailed design as a dynamic optimization problem. In addition, the solution of the screening model provides good initial guesses for the compositions and amounts of recycled batches of material for the dynamic optimization formulation. The ability to handle discrete decisions directly within the screening model makes them particularly appropriate for making decisions such as in which batch distillation region should the feed to the column be located, and what equipment should be assigned to a particular processing task.

The screening models are demonstrated on two case studies in chapters 4 and 5.

3.8 Notation

3.8.1 Indexed Sets

B The set of all batch distillation regions

B_e The set of all batch distillation regions containing fixed point e. $B_e = \{b \in B : e \in E_b\}$, so $B_e \subset B$.

E The set of all fixed points (azeotropes and pure components) in the system

E_I The set of all fixed points whose maximum composition in the product is limited (i.e., impurities), $E_I \subset E$

E_P The set of all fixed points regarded as product species, $E_P \subset E$

E_R The set of all fixed points that may be supplied to the process, $E_R \subset E$

E_b The sequence of fixed points defining the sharp splits from batch distillation region b

I The set of equipment types available in the manufacturing facility

I_D Set of equipment types suitable for distillation tasks $I_D \subseteq I$

I_R Set of equipment types suitable for reaction tasks $I_R \subseteq I$

K The set of processing trains

R_k set of reactions occurring in the reaction task in processing train k. $r = 1, \ldots N_k^R$

S The set defining the destination of the bottoms cuts $S = \{\mathrm{rxn}, \mathrm{mix}, \mathrm{waste}, \mathrm{prod}, \mathrm{any}\}$, indicating whether the bottoms are sent to the next reaction task, to the next mixing task, to waste, to product, or to anywhere in the process.

3.8.2 Integer Variables

$N_k^{batch_D}$ number of batches used for the distillation task k

$N_k^{batch_R}$ number of batches used for the reaction task k

3.8.3 Binary Variables

y_{kb}^B Is region b the active batch region for distillation k?

y_{ke}^{Bot} Is fixed point e the first fixed point appearing in the bottoms of distillation k?

y_{kin}^{C} Are n units of type i is assigned to distillation task k?

y_{k}^{merge} Is reaction task k is merged with reaction task $k+1$?

y_{ikn}^{R} Do n reactors of type i begin processing potentially merged reaction tasks at stage k?

y_{k}^{Rxn} Is reaction task k is performed?

y_{ks}^{S} Are the bottoms from distillation k are sent to s?

3.8.4 Exact linearizations of bilinear products of binary variables

z_{k}^{D} Is distillation k is performed?

z_{ikn}^{R} Are n reactors of type i are employed for reaction task k?

3.8.5 Continuous Variables

c^{equip} equipment cost for the manufacturing campaign

c^{raw} raw material cost for the manufacturing campaign

c^{total} total manufacturing cost

$c^{utility}$ utility cost for the manufacturing campaign

c^{waste} waste disposal cost for the manufacturing campaign

f_{ke}^{B} flow from splitter node k to the corresponding backward node within the expanded representation of fixed point e

$f_{ekk'}^{BF}$ time averaged flow of fixed point e from distillation k to reactors and mixers at stage k'

f_{ke}^{BI} total flow of overhead species e that could be contained in the bottoms of distillation k as an impurity

f_{ke}^{Bot} total flow of fixed point e taken in the bottoms of distillation k

$f_{kbe}^{B_{out}}$ time averaged flow of the fixed point e out of distillation k in batch region b

$f_{ke}^{D_{out}}$ time averaged flow of the fixed point e out of distillation k

f_{ke}^{F} flow of fixed point e from distillation k that is sent forward in the process for further processing

$f_{ke}^{M_{in}}$ the time averaged flow of fixed point e into mixer k

$\mathbf{f}_{k}^{M_{out}}$ the time averaged component flows into distillation k, $\mathbf{f}_{k}^{M_{out}} \in \mathbb{R}^{nc}$

$f_{e}^{Product}$ the time averaged flow rate of fixed point e in product

f_{ke}^{P} flow from splitter node k to the product node within the expanded representation of fixed point e

$f_{e}^{Prg_w}$ total flow of fixed point e purged from recycle streams that leaves the process as waste

$f_{e}^{Prg_p}$ total flow of fixed point e purged from recycle streams that leaves the process in the product stream

f_{ke}^{purge} recycled flow of fixed point e from distillation k that must be purged from the process

$f_{ke}^{R_{in}}$ the time averaged flow of fixed point e into the reactor train k

f_{ke}^{S} total flow of fixed point e into the process that is sent to reactors and mixers in processing train k

f_{e}^{Supply} the time averaged supply of fixed point e

f_{ke}^{W} flow from splitter node k to the waste node within the expanded representation of fixed point e

f_{e}^{Waste} the time averaged waste flow of fixed point e

Q_k condenser duty

$t^{campaign}$ total length of the manufacturing campaign

t_{k}^{D} processing time for distillation task k

t_{k}^{merged} total processing time for any merged reaction tasks ending with stage k

t_{k}^{R} processing time for reaction task k

$\boldsymbol{\pi}$ The barycentric coordinates, $\boldsymbol{\pi} \in \mathbb{R}^{nc}$

ξ_{kr} the extent of reaction r in reaction task k

3.8.6 Parameters

C^{cu} cost of cold utility per unit energy

C_i^E rental rate for equipment unit i

C^{hu} cost of hot utility per unit energy

C_e^r cost to purchase a unit mass of fixed point e

C_e^w cost to dispose of a unit mass of fixed point e

$f^{\max}$ upper bound for time averaged flows in the process

N_i number of equipment units i in the manufacturing facility

$N^{B^{\max}}$ maximum number of batches that may be employed during the campaign

Q^{demand} product demand

$R_i^{\min}$ the minimum reflux ratio for proper gas/liquid contacting in distillation column i

t^{charge} time required to charge one batch of material to an equipment unit

t^{empty} time required to empty one batch of material from an equipment unit

$t^{horizon}$ horizon time for manufacture

t^{reflux} time required to bring a column to total reflux

v_e underestimate of the molar volume of equilibrium point e at processing conditions

$\hat{V}_i$ processing volume of equipment unit i

V_i maximum vapor rate for distillation column $i \in I_D$

w_e molecular weight of equilibrium point e

ΔH_e^{vap} underestimate of the heat of vaporization of equilibrium point e at the at the processing conditions

$\boldsymbol{\rho}_e$ composition of fixed point e

$\boldsymbol{\nu}_{kr}$ the stoichiometric coefficients for reaction r in reaction task k

Chapter 4

Using Screening Models to Identify Favorable Processing Structures

The ability of screening models to discriminate between alternative process structures is demonstrated on a simple batch process development problem. Although only one reaction step is required in this process, the complexity of the chemistry and the thermodynamics is such that the interaction between operation of the separation and reaction tasks leads to a large set of alternative configurations for the state task network defining the process. The screening model automatically selects attractive alternatives meeting the design constraints, allowing the engineer to focus on the detailed design of these configurations. This example clearly shows the importance of quickly identifying the most attractive alternatives in order to avoid wasting time and effort optimizing designs resulting from poor synthesis decisions. Incorporating the dominant operating tradeoffs within the algebraic bounding models is the key to deriving an effective screening model for the process. This process demonstrates the type of processing tradeoffs that are important during the optimization of batch reaction/distillation networks, yet the level of detail has been minimized to highlight the specific tradeoffs exploited during the synthesis and to simplify the analysis of the resulting design.

The process examined consists of a sequence of competing first order reactions. This example also demonstrates how bounds for the extents of reaction in terms of key processing variables can be derived.

4.1 Process Description

The process examined consists of a competing set of reactions that convert the raw materials to both the desired product (P) and waste materials (W_1, W_2). The product can be separated by distillation. The bench scale synthesis employed a simple two-stage reaction/distillation process, but made use of an ice bath not available in the existing manufacturing facility. The reaction step comprises the set of competing reactions shown in (4.1). All of the reactions are first order in either A or I at the conditions under which the process may be operated. Any of the components B, W_1, or W_2 can be used to solvate the reactions.

$$\begin{tikzcd} A+B \arrow[r, "1"] \arrow[d, "2"] & I \arrow[r, "3"] \arrow[d, "4"] & P \\ W_1 & W_2 & \end{tikzcd} \tag{4.1}$$

The relative rates of the reactions have been chosen so that they agree with an early study of reaction temperature optimization (Denbigh, 1958); the reaction rates follow Arrhenius rate expressions according to the constants listed in table 4.1. All of the reactions are catalyzed by the same catalyst, and we assume that enough catalyst is present for the rate expressions to remain accurate. Degradation of the catalyst is not considered.

	k	E_A
Reaction	s^{-1}	$\frac{J}{mol}$
1	10^3	37000
2	10^7	61940
3	10^1	37000
4	10^{-3}	12058

Table 4.1: Constants for the Arrhenius rate expressions for the first order reaction rates ($r_i = C_i k_i e^{\frac{-E_A}{RT}}$).

The process considered contains the six components shown in (4.1). These components form one ternary and two binary azeotropes. The azeotropes are all contained on the facet of the composition simplex formed by B, $W1$, and P shown in figure 4-1. The composition ($\boldsymbol{\rho}_e$) of each azeotrope is shown in table 4.2. These azeotropes divide the composition space into the five batch distillation whose product sequences are shown in table 4.3.

	Azeotrope Composition		
p	W_1-P	B-W_1-P	B-W_1
B	0.00	0.72	0.35
W_1	0.15	0.06	0.65
P	0.85	0.22	0.00

Table 4.2: Azeotrope compositions for the three azeotropes formed between B, W_1, and P.

b	Product sequence
1	$\{ A, W_1\text{-}P, W_1, I, B\text{-}W_1, W_2 \}$
2	$\{ A, W_1\text{-}P, B\text{-}W_1\text{-}P, I, B\text{-}W_1, W_2 \}$
3	$\{ B, A, B\text{-}W_1\text{-}P, I, B\text{-}W_1, W_2 \}$
4	$\{ B, A, B\text{-}W_1\text{-}P, I, P, W_2 \}$
5	$\{ A, W_1\text{-}P, B\text{-}W_1\text{-}P, I, P, W_2 \}$

Table 4.3: Product cut sequences for the distillation regions.

4.2 Design Constraints

The equipment and utilities available within the manufacturing facility impose constraints on the design of the manufacturing process that often do not exist at the laboratory scale (Allgor *et al.*, 1996). Other design constraints may be imposed in order to adhere to environmental and safety regulations or to ensure the proper operation of particular tasks (i.e., temperature constraints to avoid undesirable side reactions and/or thermal runaway). These constraints must be addressed during pro-

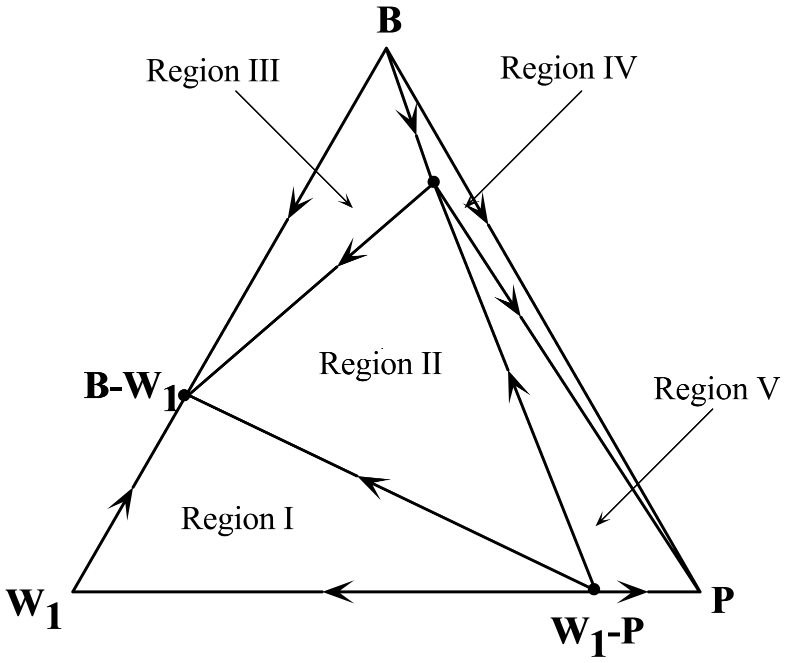

Figure 4-1: Distillation regions projected onto the facet formed by B, W_1, and P.

cess development. Imposing these restrictions may complicate the engineer's goal of rapidly designing an efficient process by requiring the engineer to focus much of his or her effort on satisfying the constraints. However, the design constraints such as emission limits, solvent to reactant ratios, conversion requirements, and temperature bounds are easily embedded within the screening models. Furthermore, these constraints are exploited during the development of the screening models themselves and actually aid in the derivation of targets for the reaction tasks.

In this example, the manufacturing facility's utility system limits the temperatures that may be employed during the operation of the tasks. Since the only cold utility is cooling water which is available at 310 K, the bench scale policy of running the reaction in an ice bath cannot be implemented in the manufacturing facility. The manufacturing facility's equipment requires that the reactions are conducted at atmospheric pressure, so the maximum reaction temperature cannot exceed either the onset temperature for thermal runaway (e.g., decomposition/polymerization) adjusted by a safety factor, or the greatest boiling temperature of any of the fixed points of the residue curve map (W_2). However, these temperature restrictions enable the derivation of bounds for the extents and selectivity of the competing reactions.

In addition, design constraints are imposed to ensure proper operation of the reactions. A molar ratio of solvent to reactant (either A or I) of at least 15 is required to ensure proper solvation of the reactions, and an excess of B (two times A) are required to maintain the desired reaction kinetics. These constraints are captured in equations (4.2–4.3).

$$\sum_e f_{ke}^{R_{in}} (\boldsymbol{\rho}_e^T \boldsymbol{\rho}_B + \boldsymbol{\rho}_e^T \boldsymbol{\rho}_{W_1} + \boldsymbol{\rho}_e^T \boldsymbol{\rho}_{W_2}) \geq 15 \left(f_{kA}^{R_{in}} + f_{kI}^{R_{in}}\right) \quad \forall k \tag{4.2}$$

$$\sum_e f_{ke}^{R_{in}} \boldsymbol{\rho}_e^T \boldsymbol{\rho}_B \geq 2 f_{kA}^{R_{in}} \quad \forall k \tag{4.3}$$

Since the product will be processed in an existing manufacturing facility, the choice of equipment is limited. The inventory and cost of the available equipment are shown in table 4.4; all of the columns contain 8 theoretical stages and must operate at a reflux ratio above 1.5 for proper gas/liquid contacting. We require that distillation columns operated in parallel at a stage are identical.

Reactors		
Volume	Available	Rental Rate
[m^3]	Units	[$ / hr]
2	1	50
3	2	70
4	1	88

Distillation Columns			
Volume	Vapor Rate	Available	Rental Rate
[m^3]	[kmol/hr]	Units	[$ / hr]
3	15	2	90
4	20	1	110
5	15	1	125

Table 4.4: Inventory and rental rates for processing equipment.

In order to evaluate the cost of manufacture, the raw material and waste disposal costs are required. In addition, in order to evaluate the utility costs and volume requirements underestimates of the heat of vaporization and the molar volume is required for all of the fixed points. These data are provided in table 4.5. Note that the waste disposal costs are merely estimates based on the average waste disposal costs for organic chemicals that are not highly toxic. Of course the most accurate data that is available should be employed, yet these figures should provide the tradeoffs similar to those that would be encountered by a manufacturer.

Fixed Points e	Raw Material [\$/kg]	Waste Removal [\$/kg]	H^{vap} [J/mol]	Molar Volume [l/kmol]	Molecular Weight
B	4.50	16.50	29300	69.210	50.08
A	7.00	16.50	35300	124.498	190.40
W_1-P		18.00	62290	196.371	240.48
W_1		18.00	40700	193.708	240.48
B-W_1-P		20.00	38080	104.759	103.39
I		18.00	45500	189.270	240.48
B-W_1		18.00	36710	150.134	173.84
P		20.00	66100	196.841	240.48
W_2		20.00	29700	194.948	240.48

Table 4.5: Material cost, disposal cost, and physical property data for the fixed points.

4.3 Reaction targets

The screening model presented in chapter 3 enforces the mass balances around the reactors in terms of the extents of the reactions. However, to capture the dominant operating tradeoffs related to the reaction tasks within the screening model, tighter bounds on the extents of reaction in terms of the operating variables must be provided. In this section, bounds for the extents of the reactions shown in (4.1) are derived in terms of the processing time and a bound on the temperature profile employed during the reaction task. These reaction targets capture key tradeoffs between the extent of reaction, selectivity, processing time, and the reactor temperature profile, yet these targets do not eliminate any portions of the feasible operating space.

4.3.1 Bounding the selectivity and extent of reaction

First, we obtain bounds on the selectivity of competing reactions. Since the selectivity of I to W_1 and the selectivity of P to W_2 depend on only the operating temperature profile, we relax the restriction that reactions 1 and 2 occur at the same temperature as reactions 3 and 4 to derive valid bounds on the selectivity. The reaction kinetics dictate that the extreme values of the selectivity are achieved at the limits of the feasible temperature range. For instance, the selectivity of reaction 1 to 2 is maximized at the minimum temperature, and the converse is true for reactions 3 and 4. Upper and lower bounds on the selectivity of the competing reactions are obtained in (4.4) and (4.5) by relating the extents of the competing reactions to the limits imposed on the operating temperature.

$$\xi_2 \frac{k_1}{k_2} e^{\frac{E_2-E_1}{RT^{\max}}} \leq \xi_1 \leq \xi_2 \frac{k_1}{k_2} e^{\frac{E_2-E_1}{RT^{\min}}} \tag{4.4}$$

$$\xi_4 \frac{k_3}{k_4} e^{\frac{E_4-E_3}{RT^{\min}}} \leq \xi_3 \leq \xi_4 \frac{k_3}{k_4} e^{\frac{E_4-E_3}{RT^{\max}}} \tag{4.5}$$

These constraints provide valid bounds on the attainable selectivity, but employ a very crude bound on the temperature profile.

Bounds for the extents of reaction in terms of the processing time are also easily derived for (4.1). Since the reaction rates are greatest at the maximum temperature of operation, the extents that can be achieved are less than the extents that would be achieved if the process operated at the maximum rate. The maximum extents of reaction are achieved when all the reactants are available at the initial time, and the reactor is operated at the maximum temperature. The solution of following differential equations defines the extents of reaction in the isothermal case:

$$\frac{d(\xi_1^{\max} + \xi_2^{\max})}{dt} = \kappa_{12}^{\max} N_A \tag{4.6}$$

$$\frac{d(\xi_3^{\max} + \xi_4^{\max})}{dt} = \kappa_{34}^{\max} N_I \tag{4.7}$$

The solution of (4.6–4.7) is defined by the following algebraic expressions relating the maximum extents of the competing reactions to the processing time when $N_A^o = f_A^{R_{in}}$ and $N_I^o = f_I^{R_{in}} \xi_1$:

$$\xi_1 + \xi_2 \leq f_A^{R_{in}}(1 - e^{-\kappa_{12}^{\max} t}) \tag{4.8}$$

$$\xi_3 + \xi_4 \leq (f_I^{R_{in}} + \xi_1)(1 - e^{-\kappa_{34}^{\max} t}) \tag{4.9}$$

where

$$\kappa_{12}^{\max} = k_1 e^{\frac{-E_{A_1}}{RT^{\max}}} + k_2 e^{\frac{-E_{A_2}}{RT^{\max}}} \tag{4.10}$$

$$\kappa_{34}^{\max} = k_3 e^{\frac{-E_{A_3}}{RT^{\max}}} + k_4 e^{\frac{-E_{A_4}}{RT^{\max}}} \tag{4.11}$$

Equation (4.9) assumes that all of the reactant I is available at the start of the reaction task in order to preserve the bounding property of the model. Note, however, that (4.8) and (4.9) are nonlinear, and that they define a nonconvex feasible region. Convex overestimates are developed for these constraints in section 4.3.2.

Equations (4.8–4.9) provide valid bounds, but they are not likely to be very tight because the constraint requiring that the same temperature determines both the selectivity and the reaction rate has been entirely relaxed. In order to tighten these bounds, we have to capture the time/temperature dependence of the operating policy within the targeting model. Incorporating the time/temperature dependence within the screening model is difficult because we are attempting to represent dynamic operating decisions using algebraic constraints. However, we can represent a bound on the feasible temperature profile using algebraic constraints. Furthermore, this representation allows us to employ the same bounds on the extents of reactions derived above. The key is to represent the total amount of time the reaction task operates within a given temperature range; we do not consider in what order the reactor spends time in each of these temperature intervals or do we require that times spent in each interval correspond to some continuous temperature profile. The feasible temperature range is divided into n_j intervals indexed by the set J. Let T_j define the maximum temperature in each interval, where $T^{\min} = T_0 < T_1 < \ldots < T_{n_j} = T^{\max}$. The time that the reaction task operates in temperature interval j is given by t_j, and the extent of reaction that is achieved in each of these intervals is specified by ξ_{krj}^T.[1] The selectivity targets previously derived are enforced over each of these temperature intervals.

$$\xi_{2j}^T \frac{k_1}{k_2} e^{\frac{E_2 - E_1}{RT_j}} \leq \xi_{1j}^T \leq \xi_{2j}^T \frac{k_1}{k_2} e^{\frac{E_2 - E_1}{RT_{j-1}}} \qquad \forall j = 1, n_j \tag{4.12}$$

$$\xi_{4j}^T \frac{k_3}{k_4} e^{\frac{E_4 - E_3}{RT_{j-1}}} \leq \xi_{3j}^T \leq \xi_{4j}^T \frac{k_3}{k_4} e^{\frac{E_4 - E_3}{RT_j}} \qquad \forall j = 1, n_j \tag{4.13}$$

The bounds on the extent of reaction that can be achieved in a given time are also enforced over each interval.

$$\xi_{1j}^T + \xi_{2j}^T \leq f_A^{R_{in}} (1 - e^{-\kappa_{12}(T_j) t_j^T}) \qquad \forall j = 1, n_j \tag{4.14}$$

$$\xi_{3j}^T + \xi_{4j}^T \leq (f_I^{R_{in}} + \xi_1)(1 - e^{-\kappa_{34}(T_j) t_j^T}) \qquad \forall j = 1, n_j \tag{4.15}$$

where

$$\kappa_{12}(T_j) = k_1 e^{\frac{-E_{A_1}}{RT_j}} + k_2 e^{\frac{-E_{A_2}}{RT_j}} \tag{4.16}$$

$$\kappa_{34}(T_j) = k_1 e^{\frac{-E_{A_3}}{RT_j}} + k_2 e^{\frac{-E_{A_4}}{RT_j}} \tag{4.17}$$

Since we do not account for the order in which the reactor spends time in each of the intervals, we have to assume that each interval is active when the concentrations are highest in order to preserve the bounding property of the screening model. Thus, we have assumed that reaction 1 occurs instantaneously when calculating the rates of reactions 3 and 4. However, the extent that can be achieved over a sequence of intervals must be less than the extent that could be achieved if the entire reaction was carried out in the last of these intervals. This is because the maximum extents are achieved over these intervals if all the raw materials are available at the initial time and the reactor operates for the duration of the time spent in all of these intervals ($\sum_{j' \leq j} t_j^T$) at the maximum temperature contained in all of these j intervals (T_j).

[1] The ξ_{krj}^T define the extent of reaction r occurring at processing stage k due to the time spent in temperature interval j. However, to simplify the notation we have dropped the subscript k throughout the following sections.

Therefore, the following constraints are also enforced.

$$\begin{aligned}\sum_{j'\leq j}\xi^T_{1j'}+\xi^T_{2j'} &\leq f_A^{R_{in}}(1-e^{-\kappa_{12}(T_j)\sum_{j'\leq j}t^T_{j'}}) \quad \forall j=1,n_j && (4.18)\\ \sum_{j'\leq j}\xi^T_{3j'}+\xi^T_{4j'} &\leq (f_I^{R_{in}}+\xi_1)(1-e^{-\kappa_{34}(T_j)\sum_{j'\leq j}t^T_{j'}}) \quad \forall j=1,n_j && (4.19)\end{aligned}$$

Constraints (4.18–4.19) are equivalent to (4.8–4.9) when the sum is taken over all of the temperature intervals (i.e., $j = n_j$); therefore, (4.8–4.9) need not be included in the optimization model. Note that (4.18–4.19) provide a tighter bound on the actual operation of the reactor than (4.8–4.9) because these constraints account for the fact that the reactions must proceed at a slower rate when not operating in the maximum temperature interval. In fact, since (4.18–4.19) are equivalent to (4.8–4.9) when $j = n_j$ and the constraints for other values of j are not necessarily inactive, (4.18–4.19) define a smaller feasible region and are tighter. The operating time for the reaction task and the extents of reaction are obtained by adding the contributions from each of the temperature intervals.

$$\begin{aligned}\sum_j t^T_j &= t \quad \forall\, j=1,n_j && (4.20)\\ \sum_j \xi^T_{rj} &= \xi_r \quad \forall\, r,\ j=1,n_j && (4.21)\end{aligned}$$

The nonlinear inequalities ((4.18)–(4.19)) and (4.14–4.15) require linear convex overestimators in order to formulate the screening model for this example as an MILP. Linear overestimates of these regions are provided in section 4.3.2.

4.3.2 Convexifying the Extent/Time Boundaries

Although the equations defining the bounds for the extents of reactions to (4.8–4.9), (4.14–4.15), and ((4.18)–(4.19)) define a feasible region that appears to be convex on first sight (the region under the surface shown in figure 4-2 appears convex), the eigenvalues of the Hessian of these functions demonstrate quite clearly that the expressions on the right hand sides of these inequalities are not concave. All of the

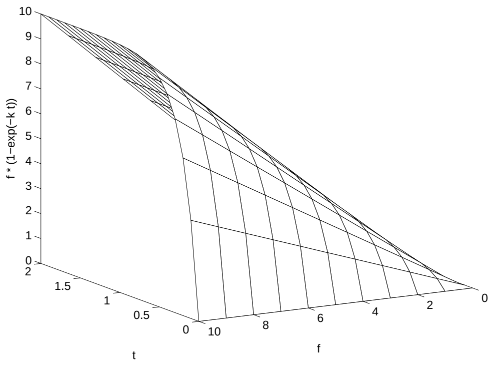

Figure 4-2: Surface defining the upper bound on the extents of reaction given by $f(1 - e^{-\kappa t})$.

expressions on the right hand side have the form $f(1-e^{-\kappa t})$ where f and t are positive variables. The Hessian of this expression is given below:

$$\mathbf{H} = \nabla^2 f(1 - e^{-\kappa t}) = \begin{bmatrix} 0 & \kappa e^{-\kappa t} \\ \kappa e^{-\kappa t} & f\kappa^2 e^{-\kappa t} \end{bmatrix} \tag{4.22}$$

The Hessian has the following eigenvalues:

$$\lambda_1 = -\frac{1}{2}e^{-\kappa t}\left(f\kappa^2 + \kappa\sqrt{f^2\kappa^2 + 4}\right) \quad \lambda_2 = -\frac{1}{2}e^{-\kappa t}\left(f\kappa^2 - \kappa\sqrt{f^2\kappa^2 + 4}\right) \tag{4.23}$$

Since the eigenvalues differ in sign, the functions defining the surface are not concave and the region under the surface is not convex. Therefore, tangents to the surface do not overestimate the function over the entire space. Examining the tangents of the surface taken at larger values of f and t shows that these planes lie above surface at all larger values of f and t, but cross the function at smaller, yet positive, values of both f and t. Examining the intersection of the tangent planes with the f-t plane shows that the line of intersection crosses through the positive orthant of the f-t plane. Two strategies have been investigated to overestimate these functions with linear constraints.

The first method defines planes that do not cut off any portions of the feasible region that are parallel to the tangent planes. Let L and M define index sets used to specify points $(\hat{f}_l, \hat{t}_{jm})^2$ at which the tangents to the function are evaluated. Hence, there exists a positive constant representing the displacement $C_{l,m}$ for each of the tangent planes that defines a parallel plane that touches the surface at only one point and will overestimate it at all other points in the feasible space ($f \geq 0$, $t \geq 0$). There exists a point ($f \geq 0$, $t \geq 0$) (the sole point of contact of the displaced plane) for which the following equation uniquely defines the constant $C_{l,m}$ that corresponds to the point $(\hat{f}_l, \hat{t}_{jm})$ at which gradient of the surface has been evaluated:

$$f\left(1 - e^{-\kappa \hat{t}_{jm}}\right) + \hat{f}_l \kappa e^{-\kappa \hat{t}_{jm}}\left(t - \hat{t}_{jm}\right) + C_{l,m} = f\left(1 - e^{-\kappa t}\right) \tag{4.24}$$

In this case, (f, t) is the sole point at which the parallel plane contacts the constraint surface. From the shape of the surface and the slope of the tangent planes, it can be seen that the single point of contact for the parallel planes is the origin. Essentially, the displacement ensures that the intersection between the tangent plane and the f-t plane does not cross the positive orthant. Setting the right hand side to zero uniquely defines the constant $C_{l,m}$ as shown below:

$$C_{l,m} = \hat{f}_l \kappa \hat{t}_{jm} e^{-\kappa \hat{t}_{jm}} \tag{4.25}$$

Displacing the tangent planes of the constraint surface by the amount $C_{l,m}$ provides linear constraints that overestimate the feasible region. This strategy can be applied to (4.18–4.19) to derive linear constraints that overestimate the feasible region. Let the sets $\hat{f}_l^A$ and $\hat{f}_l^I$ define the values of $f_A^{R_{in}}$ and $\left(f_I^{R_{in}} + \xi_1\right)$ at which the tangents to the functions appearing on the right hand sides of (4.14) and (4.15) are evaluated. The following constraints correspond to (4.14–4.15), where $\hat{f}_l^A$ and $\hat{f}_l^I$ represent fixed values of the input flows and $\hat{t}_{jm}$ is a time at which the gradients have been evaluated:

[2] The hat notation has been employed throughout this chapter to distinguish the constants used to define the interval boundaries from the subscripted variables appearing in the model.

$$\xi_{1j}^T + \xi_{2j}^T \leq f_A^{R_{in}} \left(1 - e^{-\kappa_{12}(T_j)\hat{t}_{jm}}\right) \\ + \hat{f}_l^A \kappa_{12}(T_j) e^{-\kappa_{12}(T_j)\hat{t}_{jm}} t_j^T \quad \forall j \in J, l \in L, m \in M \quad (4.26)$$

$$\xi_{3j}^T + \xi_{4j}^T \leq \left(f_I^{R_{in}} + \xi_1\right) \left(1 - e^{-\kappa_{34}(T_j)\hat{t}_{jm}}\right) \\ + \hat{f}_l^I \kappa_{34}(T_j) e^{-\kappa_{34}(T_j)\hat{t}_{jm}} t_j^T \quad \forall j \in J, l \in L, m \in M \quad (4.27)$$

A similar strategy is employed to derive a linear overestimate of the feasible region for (4.18–4.19).

The addition of these constraints does not require the introduction of any additional integer variables, but these constraints may not be very tight. In fact, these constraints do not even provide a tight approximation near the points $(\hat{f}_l^A, \hat{t}_{jm})$. Therefore, we have also considered another linearization strategy that employs additional binary variables, but leads to a tighter approximation of the nonlinear constraints.

The second linearization strategy is based on the fact that (4.14–4.15) and (4.18–4.19) define a convex feasible region if either the reagent feeds ($f_A^{R_{in}}$ and $f_I^{R_{in}} + \xi_1$) or the processing time in the given temperature interval t_j^T is fixed. Overestimating the feed flows to a particular reaction task overestimates the feasible region for all values in time. Therefore, if $f_A^{R_{in}} \leq \hat{f}_l^A$ then the tangent of $\hat{f}_l^A \left(1 - e^{-\kappa_{12}(T_j)\hat{t}_{jm}}\right)$ overestimates the original feasible region:

$$\begin{aligned} \xi_{1j}^T + \xi_{2j}^T &\leq f_A^{R_{in}} \left(1 - e^{-\kappa_{12}(T_j)t_j^T}\right) \\ &\leq \hat{f}_l^A \left(1 - e^{-\kappa_{12}(T_j)\hat{t}_{jm}}\right) + \hat{f}_l^A \kappa_{12}(T_j) e^{-\kappa_{12}(T_j)\hat{t}_{jm}} \left(t_j^T - \hat{t}_{jm}\right) \end{aligned} \qquad (4.28)$$

The extents of reactions 1 and 2 can be related to the feed of A and the fractional conversion of A. We introduce the fractional conversion of the reactants A (x^{12}) and I (x^{34}) as new variables. The fractional conversions account for the time and temperature dependence of the reactions, and the fractional conversions x^{12} and x^{34} of a batch reaction operating at temperature T_j are defined by the following concave expressions of time:

$$x_j^{12} = 1 - e^{-\kappa_{12}(T_j)t_j} \qquad \forall\, j \tag{4.29}$$

$$x_j^{34} = 1 - e^{-\kappa_{34}(T_j)t_j} \qquad \forall\, j \tag{4.30}$$

$$\tag{4.31}$$

Since (4.29) and (4.30) define concave functions of time for temperature T_j, tangents to these curves define upper bounds on the maximum conversion of A and I that can be achieved in a given temperature interval. Thus, upper bounds on x_j^{12} and x_j^{34} are defined as follows:

$$x_j^{12} \leq \left(1 - e^{-\kappa_{12}(T_j)\hat{t}_{jm}}\right) + \kappa_{12}(T_j)e^{-\kappa_{12}(T_j)\hat{t}_{jm}}\left(t_j^T - \hat{t}_{jm}\right) \qquad \forall j \in J, m \in M \tag{4.32}$$

$$x_j^{34} \leq \left(1 - e^{-\kappa_{34}(T_j)\hat{t}_{jm}}\right) + \kappa_{12}(T_j)e^{-\kappa_{34}(T_j)\hat{t}_{jm}}\left(t_j^T - \hat{t}_{jm}\right) \qquad \forall j \in J, m \in M \tag{4.33}$$

By bounding the fractional conversion according to (4.32) and (4.33), the feasible region for the extents of reaction defined in (4.14–4.15) can be overestimated using these new variables as follows:

$$\xi_{1j}^T + \xi_{2j} \leq f_A^{R_{in}} x_j^{12} \qquad \forall j \in J \tag{4.34}$$

$$\xi_{3j}^T + \xi_{4j} \leq \left(f_I^{R_{in}} + \xi_1\right) x_j^{34} \qquad \forall j \in J \tag{4.35}$$

Equations (4.34) and (4.35) both contain bilinear terms comprised of continuous variables. However, we can employ the linear expressions providing upper bounds on bilinear terms proposed by McCormick (1976), which provide the following linear upper bounds on fx:

$$fx \leq f^{\mathrm{LO}}x + fx^{\mathrm{UP}} - f^{\mathrm{LO}}x^{\mathrm{UP}} \tag{4.36}$$

$$fx \leq f^{\mathrm{UP}}x + fx^{\mathrm{LO}} - f^{\mathrm{UP}}x^{\mathrm{LO}} \tag{4.37}$$

where f^{UP} and f^{LO} provide rigorous upper and lower bounds on f; x^{UP} and x^{LO}

provide rigorous upper and lower bounds on x. The only rigorous lower bound on x_j^{12} and x_j^{34} is zero because t_j^T could equal zero, so (4.37) applied to (4.34) provides the same constraint as (4.28). However, if we can provide a nonzero bound for f^{LO}, we can employ (4.36) to derive tighter upper bounds on the extent of reaction that can be achieved.

To apply (4.36) and (4.37) bounds on the $f_A^{R_{in}}$, $f_I^{R_{in}} + \xi_1$, and on x_j^{12} and x_j^{34} are required. Upper and lower bounds on x_j^{12} and x_j^{34} of one and zero are assumed. To provide tight bounds on the feeds to the reaction tasks fixed values of the feed flows are selected so that they define an ordered set indexed by l that covers the feasible region of feed flows (i.e., $0 = \hat{f}_o^A < \hat{f}_1^A < \ldots \hat{f}_{n_l}^A = \hat{f}^{\max}$); the values of $\hat{f}_{l-1}^A$ and $\hat{f}_l^A$ can be thought to define the upper and lower bounds of a feed interval. The binary variable $y_l^{F_A}$ is introduced to identify the feed interval in which the feed lies (i.e., $\hat{f}_{l-1}^A \le f_A^{R_{in}} \le \hat{f}_l^A$). A similar set of values $\hat{f}_l^I$ and binary variables $y_l^{F_I}$ are defined for reactions 3 and 4. These binary variables represent SOS1 sets[3] and are defined by the following linear constraints:

$$\sum_{l \in L} y_l^{F_A} = 1 \tag{4.38}$$

$$\sum_{l \in L} y_l^{F_I} = 1 \tag{4.39}$$

$$\sum_{l \in L} \hat{f}_{l-1}^A y_l^{F_A} \le f_A^{R_{in}} \le \sum_{l \in L} \hat{f}_l^A y_l^{F_A} \tag{4.40}$$

$$\sum_{l \in L} \hat{f}_{l-1}^I y_l^{F_I} \le f_I^{R_{in}} + \xi_1 \le \sum_{l \in L} \hat{f}_l^I y_l^{F_I} \tag{4.41}$$

The upper ($\hat{f}_l^A$) and lower ($\hat{f}_{l-1}^A$) bounds on the flows are valid if the feed interval is active (i.e., $y_l^{F_A} = 1$), so we can derive bilinear constraints that enforce bounds on the extents of reaction that can be achieved in a given temperature interval in terms

[3] An SOS1 set is a set of binary variables with a natural ordering in which one member takes value 1 and all the others are 0. Branch and bound algorithms can take advantage of the structure of these sets during the branching procedure (Beale and Tomlin, 1970).

of the reagent feed and the time spent in the temperature interval.

$$y_l^{F_A}\left(\xi_{1j}^T + \xi_{2j}^T\right) \le \hat{f}_{l-1}^A y_l^{F_A} x_j^{12} - \hat{f}_{l-1}^A y_l^{F_A} + y_l^{F_A} f_A^{R_{in}} \qquad \forall j \in J, l \in L \tag{4.42}$$

$$y_l^{F_A}\left(\xi_{1j}^T + \xi_{2j}^T\right) \le \hat{f}_l^A x_j^{12} \qquad \forall j \in J, l \in L \tag{4.43}$$

Similar constraints can be derived for reactions 3 and 4. The exact linearization proposed by Glover (1975) can be used to transform the bilinear terms appearing (4.42) and (4.43) into an equivalent set of linear constraints.[4] To employ this strategy the variables $\tilde{\xi}_{1jl}^T = \xi_{1j}^T y_l^{F_A}$ are introduced to denote the extent of reaction 1 in temperature interval j and feed interval l. In addition, the variables $\tilde{x}_{jl}^{12} = x_j^{12} y_l^{F_A}$, $\tilde{f}_{Al}^{R_{in}} = y_l^{F_A} f_A^{R_{in}}$, and $\tilde{f}_{Il}^{R_{in}} = y_l^{F_I}(f_I^{R_{in}} + \xi_1)$ are introduced. The same procedure is applied for reactions 3 and 4. Note that $\sum_{l \in L} \tilde{\xi}_{rjl}^T = \xi_{rj}^T \ \forall\, j, r$.

Bounds on the $\tilde{\xi}_{rjl}^T$ are derived by substituting the variables for the bilinear terms into (4.42) and (4.43), yielding the following:

$$\tilde{\xi}_{1jl}^T + \tilde{\xi}_{2jl}^T \le \hat{f}_{l-1}^A \tilde{x}_{jl}^{12} - \hat{f}_{l-1}^A y_l^{F_A} + \tilde{f}_{Al}^{R_{in}} \qquad \forall j \in J, l \in L \tag{4.44}$$

$$\tilde{\xi}_{1jl}^T + \tilde{\xi}_{2jl}^T \le \hat{f}_l^A x_j^{12} \qquad \forall j \in J, l \in L \tag{4.45}$$

$$\tilde{\xi}_{3jl}^T + \tilde{\xi}_{4jl}^T \le \hat{f}_{l-1}^I \tilde{x}_{jl}^{34} - \hat{f}_{l-1}^I y_l^{F_I} + \tilde{f}_{Il}^{R_{in}} \qquad \forall j \in J, l \in L \tag{4.46}$$

$$\tilde{\xi}_{3jl}^T + \tilde{\xi}_{4jl}^T \le \hat{f}_l^I x_j^{34} \qquad \forall j \in J, l \in L \tag{4.47}$$

The constraints (4.44–4.47) overestimate the feasible region defined by the nonlinear nonconvex constraints (4.14–4.15). We can bound the region defined by (4.18–4.19) in a similar fashion. First, variables $x_j^{12^S}$ and $x_j^{34^S}$ are defined to represent the total of x_j^{12} and x_j^{34} that can be achieved in all the temperature intervals up to j:

$$x_j^{12^S} = \sum_{j' \le j} x_{j'}^{12} \qquad \forall j \in J \tag{4.48}$$

$$x_j^{34^S} = \sum_{j' \le j} x_{j'}^{34} \qquad \forall j \in J \tag{4.49}$$

[4] Section 4.6.4 discusses the linearization of the bilinear terms between continuous and binary variables.

$$x_j^{12^S} \leq \left(1 - e^{-\kappa_{12}(T_j)\hat{t}_{jm}}\right) + \kappa_{12}(T_j)e^{-\kappa_{12}(T_j)\hat{t}_{jm}} \left(\sum_{j' \leq j} t_{j'}^T - \hat{t}_{jm}\right) \qquad \forall j \in J, m \in M \tag{4.50}$$

$$x_j^{34^S} \leq \left(1 - e^{-\kappa_{34}(T_j)\hat{t}_{jm}}\right) + \kappa_{12}(T_j)e^{-\kappa_{34}(T_j)\hat{t}_{jm}} \left(\sum_{j' \leq j} t_{j'}^T - \hat{t}_{jm}\right) \qquad \forall j \in J, m \in M \tag{4.51}$$

By defining $\tilde{x}_{jl}^{12^S} = y_l^{F_A} x_j^{12^S}$ and $\tilde{x}_{jl}^{34^S} = y_l^{F_I} x_j^{34^S}$, we can derive constraints that overestimate the feasible region defined by (4.18–4.19) as follows:

$$\sum_{j' \leq j} \left(\tilde{\xi}_{1j'l}^T + \tilde{\xi}_{2j'l}^T\right) \leq \hat{f}_{l-1}^A \tilde{x}_{jl}^{12^S} - \hat{f}_{l-1}^A y_l^{F_A} + \tilde{f}_{Al}^{R_{in}} \qquad \forall j \in J, l \in L \tag{4.52}$$

$$\sum_{j' \leq j} \left(\tilde{\xi}_{1j'l}^T + \tilde{\xi}_{2j'l}^T\right) \leq \hat{f}_l^A x_j^{12^S} \qquad \forall j \in J, l \in L \tag{4.53}$$

$$\sum_{j' \leq j} \left(\tilde{\xi}_{3j'l}^T + \tilde{\xi}_{4j'l}^T\right) \leq \hat{f}_{l-1}^I \tilde{x}_{jl}^{34^S} - \hat{f}_{l-1}^I y_l^{F_I} + \tilde{f}_{Il}^{R_{in}} \qquad \forall j \in J, l \in L \tag{4.54}$$

$$\sum_{j' \leq j} \left(\tilde{\xi}_{3j'l}^T + \tilde{\xi}_{4j'l}^T\right) \leq \hat{f}_l^I x_j^{34^S} \qquad \forall j \in J, l \in L \tag{4.55}$$

Comparison of Convexification Strategies

The second strategy requires the addition of two SOS1 sets of size n_l (y^{F_A} and y^{F_I}) for each reactor included in the superstructure. The second strategy also introduces the continuous variables $\tilde{\xi}_{rjl}^T$, $\tilde{f}_{Al}^{R_{in}}$, $\tilde{f}_{Il}^{R_{in}}$, $\tilde{x}_j^{12}$, $\tilde{x}_j^{34}$, $\tilde{x}_{jl}^{12^S}$, and $\tilde{x}_{jl}^{34^S}$ which were not required for the first linearization strategy. However, the second strategy provides a tighter linearization than the first. Furthermore, the linearization provided by the second strategy can be made to approximate the original constraints as tight as is desired by increasing the sizes of the SOS1 sets. This is not possible with the first strategy.

The effort required to solve the problem given by the second linearization strategy was on the same order as the time required to solve the first. The objective calculated using the second strategy was greater than that calculated by the first, demonstrating the fact that the approximation is tighter.

The solutions that are presented in section 4.5 employ the second convexification strategy.

4.3.3 Minimum Extents of Reaction

The targets derived above capture the effects that modifications to the processing time and the temperature profile have on the selectivity and the maximum extent that can be achieved. Even though the reactions may be terminated by filtering out the catalyst, we have not placed lower bounds on the conversion that must be achieved in a given amount of time. In fact, with only these constraints, the solution of the screening model chooses to run the first two reactions to completion, separate the I, react the I to form product in the absence of W_1, and separate the product. With such a scheme, none of the product is lost in an azeotrope, making this alternative highly attractive in the screening formulation. Clearly, we would like the screening model to incorporate a lower bound on the extents of the third and fourth reactions to capture the fact that the first two reactions cannot be run to completion without producing some W_2 and P in the process. Such constraints are derived below.

A lower bound on the extent of the third and fourth reactions can be derived by underestimating both the rate of conversion of I and the amount of I that is available for reaction. The amount of I available for reaction can either be produced from the reaction of A, or it may be charged directly to the reactor. Since the reaction of I is a first order process, the extents of reactions 3 and 4 coming from each source can be treated separately; whether I is generated or charged, it obeys a first order decay, so the conversion of a given charge of I is a function of only time since the charge and the reaction temperature. Let ξ_{34}^{I} represent the extent of reactions 3 and 4 that results from I fed directly to the reactor and let ξ_{34}^{A} represent the extent of reactions 3 and 4 resulting from the conversion of A fed to the reactor.

$$\xi_3 + \xi_4 = \xi_{34}^{I} + \xi_{34}^{A} \tag{4.56}$$

Since semi-batch operation is permitted, ξ_{34}^{I} could be zero because all of the I could be charged at the end of the reaction, yet $\xi_3 + \xi_4 \geq \xi_{34}^{A}$. We focus on determining a lower bound on ξ_{34}^A. We know that ξ_{34}^A cannot be zero for nonzero values of ξ_1 because the rates of the first two reactions are finite, so the reactor operates for a period of time when I is present.

The first reaction must proceed for a certain amount of time in order to achieve a given conversion, even if the reaction proceeds at the maximum rate. As I is generated by this reaction, it immediately begins to react to form either P or W_2. The minimum extent of reactions 3 and 4 is obtained when these reactions occur at the minimum rate. Based on this observation, bounds are derived for the minimum extent of reactions 3 and 4. First, an underestimate of the time required to achieve the extent of reactions 1 and 2 is calculated. The minimum amount of time to achieve a given extent is obtained when all of the reagents are available at the initial time and the temperature is set to its upper limit, maximizing the rates. Next, an underestimate of the conversion of reactions 3 and 4 that must occur during this time is determined. To underestimate this rate, we assume that only the amount of A converted to I (i.e., ξ_1) is available at the initial time. In addition, to underestimate ξ_{34}^A we assume that all the reactions proceed at the minimum rate (i.e., the minimum temperature) for the time determined in the first step. Under these assumptions the extent of reactions 3 and 4 as a function of time can be determined from the solution of the following set of ordinary differential equations:

$$\begin{aligned}\frac{d\xi_{34}^A}{dt} &= \kappa_{34}^{\min} N_I &(4.57)\\ \frac{dN_I}{dt} &= \kappa_1^{\min} N_{\hat{A}} - \kappa_{34}^{\min} N_I &(4.58)\\ \frac{dN_{\hat{A}}}{dt} &= -\kappa_1^{\min} N_{\hat{A}} &(4.59)\end{aligned}$$

where

$$\begin{aligned}\kappa_1^{\min} &= k_1 e^{\frac{-E_{A_1}}{RT^{\min}}} &(4.60)\\ \kappa_{34}^{\min} &= k_3 e^{\frac{-E_{A_3}}{RT^{\min}}} + k_4 e^{\frac{-E_{A_4}}{RT^{\min}}} &(4.61)\end{aligned}$$

and $N_{\hat{A}}(0) = \xi_1$, $N_I(0) = 0$, and $\xi_{34}^A = 0$. Solving (4.57–4.59) subject to the initial conditions leads to the following bound on ξ_{34}^A.

$$\xi_{34}^A \geq \xi_1 \left(1 + \frac{\kappa_{34}^{\min}}{\kappa_1^{\min} - \kappa_{34}^{\min}} e^{-\kappa_1^{\min} t} - \frac{\kappa_1^{\min}}{\kappa_1^{\min} - \kappa_{34}^{\min}} e^{-\kappa_{34}^{\min} t} \right) \tag{4.62}$$

Equation (4.62) accounts for the fact that some product will be created during the reaction task as long as A is converted to I in the reactor. The region defined by (4.62) is nonconvex, yet we can provide a convex overestimate of this region by introducing an additional set of binary variables to identify a lower bound on the time required to achieve ξ_1. We enforce (4.62) for each of the temperature intervals. Discrete points in time $\hat{t}_{jm}$ are selected for each temperature interval, and the following expression for the maximum fractional conversion of reaction in this time is evaluated at each of these points:

$$\hat{x}_{jm}^{1^{\max}} = 1 - e^{\kappa_{12}(T_j)\hat{t}_{jm}} \tag{4.63}$$

At these same points in time, the minimum conversion of reactions 3 and 4 is calculated from (4.62) as follows:

$$\begin{aligned} \hat{x}_{jm}^{34^{\min}} = 1 + & \frac{\kappa_{34}(T_{j-1})}{\kappa_1(T_{j-1}) - \kappa_{34}(T_{j-1})} e^{-\kappa_1(T_{j-1})\hat{t}_{jm}} \\ & - \frac{\kappa_1(T_{j-1})}{\kappa_1(T_{j-1}) - \kappa_{34}(T_{j-1})} e^{-\kappa_{34}(T_{j-1})\hat{t}_{jm}} \end{aligned} \tag{4.64}$$

The active time interval is identified by binary variable y_{jm}^t which requires that the conversion of I achieved in temperature interval j (ξ_{1j}^T) satisfies the following constraint:

$$f_A^{R_{in}} \sum_m y_{jm}^t \hat{x}_{j,m-1}^{1^{\max}} \leq \xi_{1j}^T + \xi_{2j}^T \leq f_A^{R_{in}} \sum_m y_{jm}^t \hat{x}_{jm}^{1^{\max}} \qquad \forall\, j \tag{4.65}$$

where

$$\sum_m y_{jm}^t = 1 \qquad \forall\, j \tag{4.66}$$

A lower bound on $\xi^T_{3j} + \xi^T_{4j}$ can now be defined in terms of y^t_{jm} as follows:

$$\xi^T_{3j} + \xi^T_{4j} \geq \xi^T_{1j} \sum_m y^t_{jm} \hat{x}^{34^{\min}}_{j,m-1} \qquad \forall\, j \tag{4.67}$$

By defining the continuous variables $N^A_{jm} = y^t_{jm} f^{R_{in}}_A$ and $\xi^t_{1jm} = y^t_{jm} \xi^T_{1j}$ using an exact linearization (Glover, 1975), (4.65) and (4.67) can be expressed as the following linear constraints:

$$\sum_m N^A_{jm} \hat{x}^{1^{\max}}_{j,m-1} \leq \xi^T_{1j} + \xi^T_{2j} \leq \sum_m N^A_{jm} \hat{x}^{1^{\max}}_{jm} \qquad \forall\, j \tag{4.68}$$

$$\xi^T_{3j} + \xi^T_{4j} \geq \sum_m \xi^t_{1jm} \hat{x}^{34^{\min}}_{j,m-1} \qquad \forall\, j \tag{4.69}$$

Equation (4.69) defines a piecewise constant overestimate of the feasible region by providing a rigorous underestimate for the right hand side of (4.62).

4.4 Process Superstructure

The desired product P was synthesized at the bench scale using a process consisting of one reaction and one distillation task. During the initial phase of the reaction, the reactor was kept in at 273 K using an ice bath. After a period of time, the reactor was removed from the ice bath and heated to drive the reactions to completion. The experiments indicated that the conversion to product was affected by the time at which the reactor was removed from the ice bath. The contents remaining in the reactor at the completion of the reaction task were then separated using batch distillation.

Although the laboratory process was able to obtain P using only one reaction and distillation step, this does not imply that the optimal design of the manufacturing process should contain the same process structure. In fact, the design constraints imposed by the manufacturing facility dictate the process structure employed at the bench scale is infeasible. In order to obtain pure product, the feed to the column must lie within batch distillation regions IV and V. This requires a high selectivity of P to W_1, which implies that a high selectivity of I to W_1 must be obtained. A high selectivity of I to W_1 can be achieved when operating in an ice bath, but the selectivity is reduced at higher temperatures. The maximum selectivity that can be achieved given the cold utility available within the manufacturing facility does enable the reactor to provide a feed to the column in either region IV or V. This implies that the superstructure considered within the screening model must contain more than one reaction and distillation task to insure feasibility.

The structure of the batch distillation regions and the fact that the reactions are catalyzed by a heterogeneous catalyst also indicate that a superstructure containing more than one distillation task should be considered. Since one of the feeds to the system, B, participates in the azeotropes that are formed, it can be employed as an entrainer within the process. In addition, a stream can move from one distillation region to another through the reaction of B. Since the reactions require a heterogeneous catalyst, the reactions can be terminated by filtering out the catalyst. This indicates that it may be possible to separate the reaction mixture after a period of time, and then continue the reaction. Each of these observations indicates that a superstructure containing more that one reaction and distillation task should be considered. Two different superstructures are considered for this case study. The first superstructure contains one reaction and three distillation tasks, and the second superstructure considers three of each. Since the second superstructure contains the first, it cannot lead to a worse solution.

4.5 Solutions of the Screening Models

The cost of producing 68,039 kg of product P was minimized for both of the process superstructures mentioned above. Raw material, waste disposal, utility, and equipment rental costs were considered for a manufacturing campaign employing no intermediate storage; end effects were ignored. The product was required at a purity of 99% defined on a mass basis, and all of the bottoms streams were not permitted to be contaminated with any overhead species. Two percent of all recycled material was purged. As expected, the more flexible superstructure provided a better design and chose to employ two reaction tasks. The solutions obtained for each of the superstructures are described in sections 4.5.1 and 4.5.2. Section 4.5.3 compares the two solutions.

Five temperature intervals (defined by 310, 315, 320, 430, 440, 450 K), five feed intervals, and six time intervals were selected. The feed intervals were based on the minimum amount of A and I that is required to generate the desired amount of product at the highest selectivity possible; the upper bounds on first four intervals were given by .5, 1.1, 1.3, and 2 times this minimum amount. The bound on the final interval was given by the maximum allowable flow. A different time discretization was selected to define x^{12} and x^{34} in each temperature interval. The discrete points in time were selected to correspond to conversions of (.5, .85, .9, .99, .999, and .9999).

4.5.1 Solution obtained from the First Superstructure

The optimal solution employs one reaction and two distillation tasks. A schematic of the solution is provided in figure 4-3, where the stream labels identify the material flow in kmols for fixed points in the stream over the entire campaign. Since 345 batches are employed in this campaign, the amounts charged during each batch can be determined from the figure.

Two distillation tasks are required because a high enough selectivity of P to W_1 cannot be achieved to place the reactor effluent in either distillation region IV or V given the available cold utility. The reaction converts all of the A into products and waste materials with a small amount of I left unreacted; no A appears in the effluent. The reactor operates for 1.69 hours in the first temperature interval and for 1.5 hours in the last temperature interval. The extents of the first two reactions can be almost exclusively attributed to the time spent in the first temperature interval, and the extents of the third and fourth reactions are mostly attributed to the time spent in the last temperature interval. The reactor effluent has a composition in distillation region II, so all three azeotropes are obtained as products from the first distillation step. The $W_1 - P$ azeotrope is passed on to the second distillation step where B is

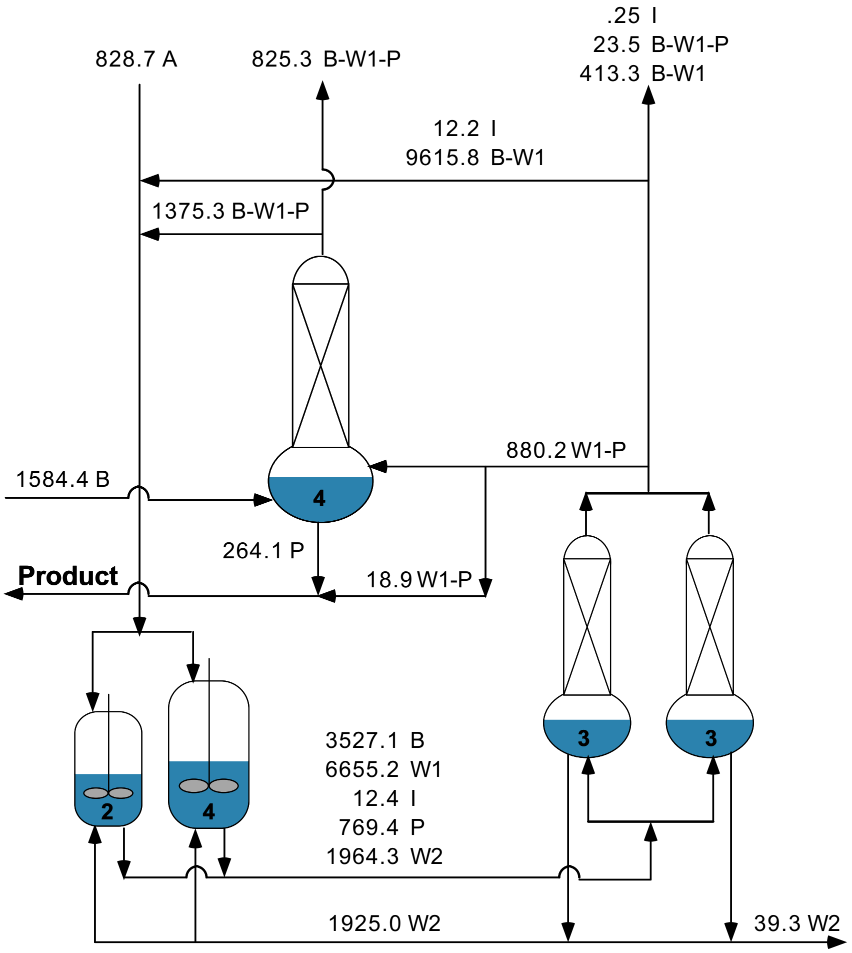

Figure 4-3: Process schematic of the solution derived from the superstructure containing only one reaction task. Fixed point flows are given in kmols.

employed as an entrainer. Enough B is added to the charge of the second distillation so that the composition of the feed lies on the boundary between distillation regions IV and V. Therefore, the only products obtained from this column are the ternary azeotrope, which is taken overhead, and the product which is taken in the bottoms of the column.

This design suffers from the fact that W_1 is only removed from the process as part of an azeotrope. As a consequence, roughly half of the B fed to the process leaves as waste, and over 40 % of the P that is generated is lost in the ternary azeotrope. Not surprisingly, the waste disposal costs dominate the production costs for this design, as shown in table 4.10. Tables 4.6, 4.7, and 4.8 show the material processing costs for the campaign. Table 4.9 shows the charges incurred for the use of equipment during the campaign. The 2 and 4 m^3 reactors are employed for the reaction step, both 3 m^3 columns are employed for the first distillation, and the 4 m^3 column is used for the second distillation. The batch size and cycle time are limited by the first reaction and distillation tasks.

Raw Material Costs				
Raw Material	Cost [$/kg]	Feed [kg]	Total Cost [$]	$ / kg product
B	4.50	79347.09	357061.89	5.25
A	7.00	157787.64	1104513.51	16.23
Total		237134.73	1461575.40	21.48

Table 4.6: Raw material costs for the design obtained from the first superstructure.

Waste Disposal Costs				
Waste Material	Cost [$ / kg]	Amount [kg]	Total Cost [$]	$ / kg product
B-W1-P	20.00	87746.25	1754924.95	25.79
I	18.00	59.81	1076.50	0.02
B-W1	18.00	71842.50	1293164.95	19.01
W2	20.00	9447.23	188944.61	2.78
Total		169095.78	3238111.01	47.59

Table 4.7: Waste disposal costs for the design obtained from the first superstructure.

Utility Costs			
Cut Material	Amount [kg]	Reboiler Cost [$]	$ / kg product
Distillation 1			
W1-P	216212.66	443.97	0.01
B-W1-P	2424.69	7.08	0.00
I	2990.28	4.49	0.00
B-W1	1743455.39	2918.63	0.04
Distillation 2			
B-W1-P	227520.80	664.30	0.01
Total	2192603.82	4038.46	0.07

Table 4.8: Utility costs for the design obtained from the first superstructure.

Reactor Rental Costs					
Volume		Assigned	Rental Rate	Total Cost	$ per
[gal]		Units	[$ / hr]	[$]	kg product
2		1	50	71558.56	1051.73
4		1	88	125943.07	1851.04
Distillation Column Rental Costs					
Volume	Vapor Rate	Assigned	Rental Rate	Total Cost	$ per
[gal]	[kmol/hr]	Units	[$ / hr]	[$]	kg product
3	15	2	90	257610.82	3786.23
4	20	1	110	157428.83	2313.80
Total for reactors and columns				612541.28	9.00

Table 4.9: Equipment costs for the design obtained from the first superstructure.

Cost Contributions			
Component	Percent	Total Cost [$]	$ / kg product
Raw Material	27.49	1461575.40	21.48
Waste Disposal	60.90	3238111.01	47.59
Utility	0.09	5048.08	0.07
Equipment	11.52	612541.28	9.00
Total		5317275.78	78.15

Table 4.10: Comparison of raw material, waste disposal, utility, and equipment for the design obtained from the first superstructure.

Utilization Measure	Processing Task		
	Reaction 1	Distillation 1	Distillation 2
Cycle Time	4.15	4.15	2.30
Volume Required	6.00	6.00	0.82
Volume Assigned	6.00	6.00	4.00

Table 4.11: Equipment utilization for the design obtained from the first superstructure.

4.5.2 Solution obtained from the Second Superstructure

The optimal solution obtained from the second superstructure employs two distillation and two reaction tasks. A schematic of the solution is provided in figure 4-4 in which the streams are labeled with the flow of material in kmols for the entire campaign specified in terms of the fixed point flows. Since 233 batches are employed in this campaign, the amounts charged during each batch can be determined from the figure.

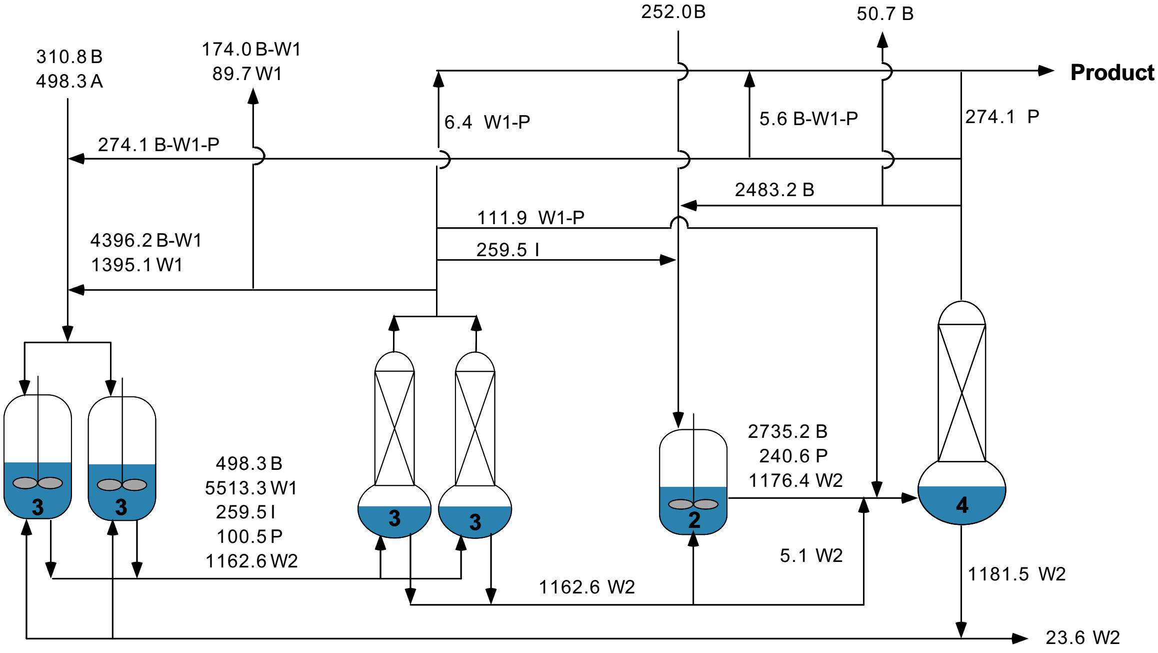

Figure 4-4: Process schematic of the solution derived from the superstructure permitting multiple reaction tasks. Fixed point flows are given in kmols.

The solution obtained from this superstructure exploits the fact that the reactions can be terminated by filtering the heterogeneous catalyst from the reacting mixture. In the absence of the catalyst, the mixture can be separated by batch distillation without the reaction continuing as the distillation is performed. The first reaction task is run to complete conversion of A, but only a portion of the generated I is converted through the third and fourth reactions. The conversion achieved by the first two reactions can be attributed to the time spent in the first temperature interval. At these low temperatures a high selectivity of I to W_1 is achieved. The extents of the third and fourth reactions is kept relatively small; these extents must be large enough to satisfy the minimum conversion constraints which are active for the first temperature interval. However, most of the conversion obtained for the third and fourth reactions can be attributed to time spent in the last temperature interval in which a high selectivity of P to W_2 is achieved. Enough time was spent in the first interval to achieve total conversion of A at high selectivity.

Stopping the second reaction task after a limited conversion was achieved in reactions 3 and 4 allows the separation to be performed in the presence of less product. A large quantity of W_1 is employed as a solvent for the first reaction task, placing the composition of the reactor effluent in batch distillation region I. This enables the first distillation task to obtain pure W_1 in one of the cuts, permitting W_1 to leave the system in pure form. The intermediate is passed on to the second reaction task for conversion into the desired product. The second reaction task operates at the highest allowable temperature in order to achieve both fast reaction rates and a high selectivity of P to W_2. Note that a large amount of B is employed as a solvent in this reaction step. The effluent from this reaction task is combined with the $W_1 - P$ cut from the first distillation to place the feed to the second column in batch distillation region IV. On first sight, the use of B as a solvent for the second reaction task seems peculiar. However, the solvent requirements were specified on a mole basis, and B has a smaller molar volume than W_2 (the other potential solvent). The equipment cost savings achieved by using B instead of W_2 and employing a smaller reactor outweigh the separation cost incurred by taking the B overhead instead of taking W_2 in the bottoms.

This design makes fairly efficient use of both the raw materials and the available equipment. The only way that reactants and products leave the process as waste is through the purge of recycled streams. A more detailed summary of the material processing costs is provided by tables 4.12, 4.13, and 4.14. The equipment items are all running at or near capacity, except for the column assigned to the second distillation task. Table 4.15 shows the charges incurred for the use of equipment during the campaign, and table 4.16 shows the utilization of the equipment items.

Raw Material Costs				
Raw Material	Cost [$ / kg]	Feed [kg]	Total Cost [$]	$ / kg product
B	4.50	28191.31	126860.91	1.86
A	7.00	94867.93	664075.49	9.76
Total		123059.24	790936.40	11.62

Table 4.12: Raw material costs for the design obtained from the second superstructure.

Waste Disposal Costs				
Waste Material	Cost [$ / kg]	Amount [kg]	Total Cost [$]	$ / kg product
B	16.50	2537.91	41875.60	0.62
W1	18.00	41850.19	753303.42	11.07
B-W1	18.00	4949.53	89091.57	1.31
W2	20.00	5682.65	113653.07	1.67
Total		55020.29	997923.66	14.67

Table 4.13: Waste disposal costs for the design obtained from the second superstructure.

Utility Costs			
Cut Material	Amount [kg]	Reboiler Cost [$]	$ / kg product
Distillation 1			
W1-P	28431.31	58.38	0.00
W1	1099039.58	1474.56	0.02
I	62407.76	93.61	0.00
B-W1	247476.60	414.29	0.01
Distillation 2			
B	126895.75	588.55	0.01
B-W1-P	28916.36	84.43	0.00
P	65931.99	143.67	0.00
Total	1659099.35	2857.48	0.05

Table 4.14: Utility costs for the design obtained from the second superstructure.

Reactor Rental Costs					
Volume		Assigned	Rental Rate	Total Cost	$ per
[gal]		Units	[$ / hr]	[$]	kg product
2		1	50	44023.00	647.03
3		2	70	123264.40	1811.67
Distillation Column Rental Costs					
Volume	Vapor Rate	Assigned	Rental Rate	Total Cost	$ per
[gal]	[kmol/hr]	Units	[$ / hr]	[$]	kg product
3	15	2	90	158482.80	2329.30
4	20	1	110	96850.60	1423.46
Total for reactors and columns				422620.81	6.21

Table 4.15: Equipment costs for the design obtained from the second superstructure.

Utilization	Processing Task			
Measure	Reaction 1	Distillation 1	Reaction 2	Distillation 2
Cycle Time	3.78	3.78	3.78	3.16
Volume Required	6.00	6.00	2.00	2.10
Volume Assigned	6.00	6.00	2.00	4.00

Table 4.16: Equipment utilization for the design obtained from the second superstructure.

Cost Contributions			
Component	Percent	Total Cost [$]	$ / kg product
Raw Material	35.71	790936.40	11.62
Waste Disposal	45.05	997923.66	14.67
Utility	0.16	3571.85	0.05
Equipment	19.08	422620.81	6.21
Total		2215052.72	32.56

Table 4.17: Comparison of raw material, waste disposal, utility, and equipment costs obtained for the second superstructure.

4.5.3 Solution Comparison

The solution obtained from the second superstructure produces a much more efficient design. This is primarily due to the fact that the waste material W_1 formed during the reactions can be removed in pure form in the second case, but not in the first. This results in much lower raw material and waste costs. The difference in the equipment costs result from the fact that the first superstructure requires a much longer campaign, since it obtains much less product for each batch that is processed. A comparison of the cost contributions between the two campaigns is given in table 4.18.

Cost Component	First Superstructure [$ / kg Product]	Second Superstructure [$ / kg Product]
Raw Material	21.48	11.62
Waste Disposal	47.59	14.67
Utility	0.07	0.05
Equipment	9.00	6.21
Total	78.15	32.56

Table 4.18: Comparison of the manufacturing costs of the solutions obtained from the two superstructures examined.

4.6 Computational Considerations

The screening models presented in this chapter are formulated as mixed-integer linear programs. Although the global optimum of such models can be found using standard algorithms, the solution time may be prohibitive. For these types of problems, strong formulations are required in order to attempt to solve large problems. In addition, the ability of the linear programming and branch and bound algorithms to solve these models reliably requires that the model is well-scaled. Although the focus of this research has not been to derive the strongest equivalent formulations for these models, the procedure used to solve these models can dictate whether solution is possible in a reasonable time using standard MILP solution codes. In this section the techniques that have been employed to permit the solution of the screening model are discussed. Specifically, the modifications required to provide a well-scaled model, the procedure employed to reduce the size of the MILP and obtain tighter bounds on the continuous variables involved in bilinear terms, and the linearization method employed for the bilinear terms are described.

4.6.1 Size of the Models solved

The screening models solved within this thesis are fairly large, and can be difficult to solve. The following sections cover some of the techniques that have been employed to solve these models in a reasonable amount of time. Table 4.6.1 provides statistics about the size of the models involved in the case studies presented in chapters 4 and 5. Note that the number of binary variables reported treats each SOS1 set as one binary variable; this means that an SOS1 set comprised of five binary variables (e.g., the variable $y_l^{F_A}$ in chapter 4) is counted as only one binary variable rather than five. For reference, the number of SOS1 sets has been included in the table. The solution times reported for the models are given to provide a rough idea of how long the models take to solve.[5] The solution times depend on what type of machine on which the models

[5] The case study from chapter 4 containing only one reaction task contains more variables and constraints than the superstructure containing two reaction tasks because more batches were permit-

Case Study	Binary Variables	SOS1 Sets	Continuous Variables	# of Constraints	Approximate Solution Time
Chapter 4: One Rxn	47	8	3662	6512	2.4 hrs
Chapter 4: Two Rxns	48	9	2612	4712	2.5 hrs
Chapter 5: Case I.A	98	10	2104	3046	3.5 hrs
Chapter 5: Case I.B	98	10	2097	3035	30 min
Chapter 5: Case II	32	11	2061	3574	25 min
Chapter 5: Case III	32	11	3196	5861	40 min

Table 4.19: Size and approximate solution times for the screening models solved in chapters 4 and 5 on an HP J200 workstation.

were solved and what other jobs were running on the machine. All the models were solved using OSL (IBM, 1991) within GAMS (Brooke *et al.*, 1992).

4.6.2 Scaling of the Linear Programs

The model described in the preceding sections can lead to linear programs that are sufficiently poorly scaled to cause the simplex codes to fail due to numerical problems; such problems were encountered within both OSL (IBM, 1991) and CPLEX (CPL, 1993). The poorly scaled LPs are the result of nonzero elements of the constraint matrix that vary over many orders of magnitude. In many situations, such problems result from a poor choice of units for the modeling variables (analogous to the column/variable scaling discussed in chapter 7). However, poorly scaled models can also be the result of modeling decisions such as whether certain tradeoffs are important or not.

The scaling problems within these models come from the linearized constraints employed to bound the conversion of reactants with respect to time and temperature such as those appearing in (4.32) and (4.33). When the terms $e^{-\kappa(T_j)\hat{t}_{jm}}$ become very small, these constraints are very poorly scaled because the coefficient for the extents are unity, but the coefficient of time is a nonzero value that is approaching zero with larger values of $\hat{t}_{jm}$. In order to avoid these scaling problems, a different time

ted in the one reaction case. The number of batches is represented by an SOS1 set that is involved in bilinear terms, so the number of variables and equations is larger.

discretization was selected for reactions 1 and 2 and reactions 3 and 4 in each time interval. The times were selected to correspond to conversions that were different from unity by at least the optimization tolerances. If we had selected only one time grid, then we could ignore these constraints for values of $e^{-\kappa(T_j)t_m}$ below some threshold. This threshold value indicates the point in time at which the slope of $1 - e^{-\kappa(T_j)t}$ is small enough to be ignored. Eliminating these constraints makes the model well-scaled. However, the elimination of these constraints defines a threshold time beyond which total conversion can be achieved, whereas in reality total conversion is never achieved. We have found that both approaches lead to a well scaled model, but have chosen to employ different time discretization for each temperature interval in the examples considered in this chapter.

4.6.3 Solution Procedure

A sequence of simpler models is solved before the full screening model is solved. These simpler models are solved for three main reasons: 1) to obtain tighter bounds on the continuous variables that are involved in the bilinear expressions appearing in the model, and 2) to reduce the size of the MILP that is attempted, and 3) to determine a feasible assignment of a large number of the integer decision variables, permitting an incumbent solution to be found with little additional effort.

In the sequence of models that is solved, the number of integer variables appearing in the model is increased. By solving the simpler models first, a feasible value of the integer variables for the larger problem can be determined with little additional effort. For example, first the cost of raw material and waste disposal costs is minimized using simple bounds for the reaction selectivity that assume only one temperature interval and no bounds on the extents of reaction versus time. The location of the bottoms cut is not defined and processing times are not considered. In this model, the only binary variables that appear are those defining the active batch distillation region and those identifying whether the reaction tasks are performed. This model can be solved quickly. The binary variables from the optimal solution are then fixed, and a more complicated model that includes the definition of the bottoms is then solved for the same objective function. The solution to this problem provides what is hoped to be a good solution, but probably not optimal. All of the integer variables are then set free, and the problem is solved again. However, the solution just obtained for this model is provided to the optimizer and is used to prune the branch and bound tree. All branches with solutions worse than this value (the incumbent) are not examined. The incumbent value could also be determined using heuristic methods. In fact, good heuristic methods may provide better incumbent solutions. However, as we discuss in the next paragraph, some of the simple models must be solved to global optimality, since we employ their solution to provide rigorous bounds on parameters appearing in the model.

Another reason for solving the simple models is to provide tighter bounds on parameters appearing in the screening model that are used to linearize the bilinear expressions, or to reduce the size of the screening model. For instance, the minimum campaign length is used in the linear expressions defining the time that each equipment item is employed. While we have found that solving for the minimum campaign length is more difficult than solving the screening model, we can obtain a lower bound on the minimum campaign length by solving two simpler problems. We determine both the minimum number of batches that is required to meet the production demands and a lower bound on the processing time for the distillation tasks. If we ignore the equipment allocation constraints, a lower bound on the minimum distillation processing time can be determined from the amount of material taken overhead in the distillation columns. This bound may not be very tight since the same distillation columns can be used for all of the distillations, yet it tightens the linearization of the bilinear terms, improving the efficiency of the branch and bound procedure. Similarly, determining the minimum number of batches serves two purposes: it defines a lower bound for the campaign length when used in conjunction with the lower bound on the distillation processing time, and it allows the size of the MILP to be reduced. The number of batches is represented using an SOS1 set, i.e., $N^{batch} = \sum_{nb} y_{nb}^{NB}$. Some of the constraints that are generated result from linearizing bilinear terms involving y^{NB}. These constraints are only generated for values of nb

that are greater than or equal to the minimum number of batches; for values of nb that are less than the minimum, any feasible solution has $y_{nb}^{NB} = 0$, and the corresponding constraints are inactive. Therefore, these constraints can be safely eliminated.

The sequence of models that is solved is listed below, with a short description of the reason for solving each model.

Material: This model determines a lower bound on the raw material and waste disposal cost for the manufacturing campaign. Simple bounds on the selectivity are imposed. No dependence on time is considered. The solution provides a lower bound on the raw material and waste costs and identifies the active batch distillation regions.

Bottoms: This model identifies the location of the bottoms cuts and minimizes the raw material, utility, and waste disposal costs. The targets for the extents of reaction described in this chapter are employed. The utility cost that is calculated represents a lower bound on the utility cost determined by the full screening model, because the minimum reflux ratio of all of the columns that are available is employed to calculate the utility costs.

Distillation Time This model determines a lower bound on the total processing time required for the distillation tasks. The model is first solved with the binary variables fixed at the solution of the Bottoms model to provide an incumbent solution. The model is then solved to optimality with all of the binary variables remaining free.

Batches The minimum number of batches is determined. This model determines a feasible allocation of the equipment units that minimizes the number of batches required. First, the model is solved with the location of the distillation cuts held fixed, providing an upper bound on the optimal solution. Next, a relaxed model is solved. The solutions of these two models provide upper and lower bounds on the minimum number of batches and are used to reduce the size of the Batches model. Finally, the model is solved to optimality. The optimal solution of Batches is used to reduce the size of the screening model. Note that the fact that the number of batches is an integer value can be exploited when determining the termination criteria of the branch and bound algorithm. The solution also provides a lower bound on the campaign cost when combined with the solution of the Distillation Time model.

Units The minimum number of equipment units required to manufacture the product is determined. This quantity has been employed to tighten the constraints defining the time that the equipment units are used which result from the exact linearization of the bilinear expressions involving the campaign length and the SOS1 variables denoting how many equipment items of a particular type are employed (see section 4.6.5).

Screening Model This model minimizes the equipment, utility, raw material, and waste disposal costs. The values of the integer values determined from the solution of Units and Batches can be employed to quickly solve the Screening Model to obtain an upper bound on the solution. The smallest of these can be employed as an incumbent. Heuristics can also be employed to define an incumbent solution, but this has not been investigated in any detail. However, the screening model can be solved quickly when the allocation of the equipment items is fixed, so this could be exploited in deriving a heuristic procedure to specify the incumbent.

4.6.4 Linearization of Bilinear Terms

The screening model that has been presented has been written in a form that contains only binary and continuous variables. The integer variables in the model have been replaced by binary variables; for example, $N^B = \sum_{n=1}^{N} y_n^N n$. However, the model originally contained bilinear expressions that have been eliminated through the introduction of additional continuous variables and constraints to cast the model as a MILP. Since all of the bilinear terms in the original model are between two binary variables or between a binary and a continuous variable, an exact transformation exists and has been employed. Although several ways in which to generate linear constraints defining an equivalent convex hull of integral solutions exist, the choice of the linearization technique can have a major impact on the strength of the formulation (the way in which the relaxed problem approximates the convex hull). We have applied ideas developed in the operations research community to carry out this transformation in a systematic fashion, employing the method leading to the strongest formulation whenever the choice between the methods was clear. We have not considered algorithms designed to deal directly with the bilinear models (Quesada and Grossmann, 1995; Al-Khayyal, 1992) although we recognize that research in this area may enable these models to be solved more efficiently. We have employed the techniques of Glover (1974; 1975) and Adams and Sherali (Adams and Sherali, 1986; Adams and Sherali, 1990; Adams and Sherali, 1993) to transform the original bilinear expressions into linear inequalities.

First, we show the way that the bilinear terms in the model can be replaced with new continuous variables that equal the original bilinear expression for all integer values of the binary variables. The screening model contains bilinear terms between two binary variables, or between a binary and a continuous variable. An exact linearization for each type of expression was proposed by Glover (1975). Let $x \in [x^{LO}, x^{UP}]$ and $y_1, y_2 \in \{0, 1\}$ represent the continuous and binary variables involved in the bilinear terms xy_1 and y_1y_2. Continuous variables $z^C = xy_1$ and $z^B = y_1y_2$ are introduced to replace these terms. The following inequalities (Glover, 1975) define z^C:

$$x - x^{UP}(1 - y_1) \leq z^C \leq x - x^{LO}(1 - y_1) \qquad (4.70)$$

$$x^{LO}y_1 \leq z^C \leq x^{UP}y_1 \qquad (4.71)$$

and the following inequalities define z^B (Glover and Wolsey, 1974):

$$z^B \leq y_1 \qquad (4.72)$$

$$z^B \leq y_2 \qquad (4.73)$$

$$z^B \leq y_1 + y_2 - 1 \qquad (4.74)$$

However, (4.70–4.74) only define z^C and z^B exactly when y_1 and y_2 take integer values. Since the binary variables are relaxed during the solution of the MILP, how well these constraints approximate the convex hull is important. The values chosen for x^{LO} and x^{UP} have a major impact on the way in which (4.70–4.71) affect the integrality gap of the problem.[6] A poor choice of x^{LO} and x^{UP} will lead to a loose LP relaxation. These models may be solved more efficiently if tight bounds on the continuous variables involved in the bilinear expressions can be derived. The solution procedure that we have proposed attempts to derive tight bounds for these quantities, but we recognize that these constraints have a negative impact on the performance of the solution algorithms.

The work of Adams and Sherali (1986; 1990; 1993) addresses the strength of the formulation resulting from the exact linearization of bilinear terms involving binary variables. They address mixed-integer zero-one quadratic programming problem (MIQPP) and mixed integer bilinear programming problems (MIBLP). MIQPP and MIBLP problems can be reformulated using one of several exact linearization methods (Adams and Sherali, 1990; Adams and Sherali, 1993). The different linearization schemes affect the number of constraints in the resulting mixed integer zero-one linear program and the tightness of the linear programming relaxation. The linearization technique proposed by Adams and Sherali (1990) has been shown to theoretically dominate previously proposed linearization techniques (Glover and Wolsey, 1974; Glover, 1975) for MIQPP problems. However, this technique results in a larger number of constraints. They also propose an efficient solution algorithm for the MIBLP problems (Adams and Sherali, 1993).

Their technique generates a tight linear reformulation for mixed-integer zero-one programming problems. The original constraints in the problem are multiplied by every binary variable to derive an additional set of nonlinear constraints. The constraints involving only binary variables are multiplied by the differences between the continuous variables and their bounds (e.g., $x^{UP} - x$ and $x - x^{LO}$). Continuous variables are then introduced to represent the bilinear terms using the same linearization

[6] Sometimes constraints in this form are referred to as ‘Big M’ constraints.

scheme proposed by Glover (1975), resulting in a mixed-integer linear model.

Unfortunately, the screening model developed in the preceding chapter is not in MIQPP or MIBLP form; MIQPP and MIBLP models require that all of the bilinear terms in the model appear in the objective function. All of the bilinear terms defining costs, (3.71–3.74), can be moved into objective function, but the remaining bilinear terms in the screening model cannot be directly moved to the objective function.

Noting that the techniques developed by Adams and Sherali lead to a tighter formulation, but do not apply directly to our problem, we have applied their ideas in the following fashion. First, we employ the exact linearization proposed by Glover (1975) to generate an exact linearization of all of the bilinear terms originally appearing in our model. Next, we apply the basic idea proposed by Adams and Sherali (1986; 1990) in a limited sense. We look at the set of new continuous variables that we have introduced and multiply any equations containing only binary variables by the difference between the continuous variables and their bounds or by other binary variables if these multiplications will not introduce any additional continuous variables. We multiply the other constraints by any binary variables that will not introduce any additional continuous variables due to new bilinear terms. This idea was carried out manually, so new equations that could have been introduced may have been missed. The application of the idea presented above seems to have the biggest impact when the SOS1 variables were involved in bilinear expressions. For example, consider the bilinear term $y_n f = f_n$ where $\sum_n y_n = 1$ and $f \in [0, f^{UP}]$. The application of the procedure results in $\sum_n (y_n f^{UP} - f_n) = f^{UP} - f$ which reduces to $\sum_n f_n = f$. Although these constraints are somewhat obvious from a physical understanding of the system, they are derived by this procedure. Although other constraints were derived and added to the model, the biggest impact on the efficiency seemed to come from the constraints involving the SOS1 variables.

To compare the benefits of the different linearization strategies effectively, the transformations from the bilinear model to the different equivalent linear representations must be performed automatically. This was not attempted because the proposed models could be solved with the strategy that was applied. However, if the solution of much larger models is attempted, automatic derivation of a tighter equivalent linear model may be required. With different strategies implemented automatically, the tradeoff between model size and solution efficiency can be investigated empirically.

4.6.5 Influencing the Branch and Bound Algorithm

Features of the models have been exploited to improve the performance of the branch and bound procedure. These include the identification of SOS1 sets and the use of variable priorities.

Many of the binary variables in the system represent special ordered sets of type 1 (SOS1) (Beale and Tomlin, 1970), such as the number of batches and the type of distillation column assigned to a distillation task. Declaring these variables as SOS1 sets allows the branch and bound algorithm to employ a different branching procedure for these sets. Typically, during the branching procedure, the variables in the set are divided into subsets in which one subset contains the nonzero element and the other does not. This differs from the usual practice of fixing a binary variable to either zero or one along each branch, and is much more efficient when the SOS1 sets contain many elements. For small sets, the benefits may not be very pronounced. In addition, the fact that these variables must sum to one helps when linearizing the bilinear terms between the SOS1 and continuous variables. This is explained in section 4.6.4.

Since some of the decisions in the design of the process are naturally made in a sequential fashion, this sequence can be used to indicate a preferred branching order for the branch and bound algorithm. For instance, there is no point in deciding which distillation column to assign to a separation task if the separation is not performed. The same holds for the reaction tasks. Variable priorities are a way to represent the preferred branching order to the solvers embedded within GAMS (Brooke *et al.*, 1992). Empirical evidence has also suggested the addition of an SOS1 set to represent the number of items of a particular equipment item assigned to the process. When this set is employed in conjunction with the setting of priorities, the branch and bound decides whether to employ a particular item of equipment before determining where to assign the unit. Experience solving these models has shown that the following ordering of the discrete decisions (from top to bottom in the tree) improves the performance of the algorithm:

1. the existence of the reaction tasks
2. the existence of the distillation tasks
3. the identity of the active batch regions
4. the number of distillation columns assigned to a particular separation
5. the location of the bottoms cuts
6. what equipment units are employed within the process
7. the allocation of reactors and columns to particular tasks
8. identifying the active feed and time intervals
9. determining the number of batches

4.6.6 Tailored Solution Procedures

This research has not investigate tailored solution procedures for the solution of the screening models. However, it is easy to recognize that a tailored solution procedure would be more effective on the screening models, particularly one that can exploit the way in which the number of batches has been modeled. For the models with no intermediate storage, all units employ the same number of batches, so the number of batches has been represented using a single SOS1 set. The size of this set affects the number of equations in the model to be solved. In addition, the upper and lower bounds on the number of batches appear in the constraints used to linearize the bilinear expressions involving the number of batches and any continuous variables (e.g., terms defining the charge of material to a particular task for each batch). Thus, by restricting the number of batches to several smaller ranges, not only can the size of the model in each range be reduced, but each of these models will result in a tighter formulation since the tighter upper and lower bounds can be employed.

A tailored branch and bound procedure could reduce the size of the models and update the parameters when branching on members of the SOS1 set. Although the implementation of such a procedure is a nontrivial task, it may be required to handle situations employing unlimited intermediate storage, or cases in which intermediate storage is employed to decouple only some of the processing trains. In these situations, the screening model includes a number of batches for each processing step.

4.6.7 Representation of Batch Distillation Boundaries

The boundaries of each of the product simplices are included in the each of the batch distillation regions. Thus, if the feed to a distillation column is located on a boundary, two choices of the binary variables lead to exactly the same solution. This requires the branch and bound procedure to search each of the trees to verify the solution. These situations are common and will almost always arise from the addition of an entrainer. For instance, in the solution to the superstructure containing only one reaction task, both distillation tasks have feeds located on the boundary located between two distillation regions. Future work should investigate ways to avoid this type of problem.

4.7 Summary

The application of the screening models to a fairly simple process has been examined. This chapter demonstrates how the design constraints and the restrictions imposed by the manufacturing facility can be used to derive bounds for the extent of reaction versus time and for the selectivity of competing reactions. However, even for a reasonably simple problem, the derivation of these bounds may be a nontrivial task.

The application of the bounds to two different superstructures demonstrates that even rough approximations of the reaction behavior can capture many of the tradeoffs that need to be considered at the design stage. In fact, solution of the screening model may exploit tradeoffs that are not obvious to the engineer. In some cases, these solutions may indicate that the screening model should be augmented with additional constraints to capture some particular physical behavior that was relaxed during the derivation of the screening model. For example, the solution chose not to perform any of the third and fourth reactions in the one of the reactors until a constraint requiring a minimum conversion with respect to the reaction time was added. In other cases, the solution of the screening model may generate design alternatives that differ substantially from the designs produced through minor modifications of the chemists recipe. In retrospect, the solution determined from the second superstructure seems obvious. However, if we had started with the mindset of adapting the chemists design to account for the fact that we could not operate at such a low temperature, we may have ended up with a design looking much more like the one obtained from the first superstructure.

The difference in the solution obtained from the two superstructures demonstrates the need to consider a broad range of alternatives early in the design of the process. This highlights both a strength and a weakness of the screening models in the example presented. First, by only including a subset of the constraints the models do not eliminate any promising designs contained within the superstructure. However, since only reaction/distillation processes are included within the current superstructure, many batch processes of interest cannot be described by the screening models described here. Thus, targeting models for other common processing tasks such as extraction, crystallization, etc. should be investigated in the future.

4.8 Notation

The notation that has been introduced in this chapter is defined in the lists below.

4.8.1 Indexed Sets

J The set defining the temperature intervals. For $j \in J$, T_{j-1} and T_j represent the lower and upper bounds of the interval.

L The set defining the feed intervals. For $l \in L$, $\hat{f}_{l-1}$ and $\hat{f}_l$ represent the lower and upper bounds of the interval.

M The set defining the time intervals. For $m \in M$, t_{m-1} and t_m represent the lower and upper bounds of the interval. Note that $t_0 = 0$.

4.8.2 Binary Variables

$y_l^{F_A}$ SOS1 set denoting the active feed interval for the A charged.

$y_l^{F_I}$ SOS1 set denoting the active feed interval for the I charged and the I generated by reaction 1.

y_{jm}^t SOS1 set denoting the active time interval in temperature interval j.

4.8.3 Variables

N_{jm}^A the amount of A available for reaction in the time interval m and temperature interval j. $N_{jm}^A = y_{jm}^t f_A^{R_{in}}$.

ξ_{1jm}^t Continuous variable representing a bilinear product between the following continuous and binary variables $\xi_{1jm}^t = y_{jm}^t \xi_{1j}^T$.

ξ_{rj}^T the extent of reaction r attributed to temperature interval j

$\tilde{\xi}_{rjl}^T$ the extent of reaction r attributed to temperature interval j and feed interval l. Note $\sum_l \tilde{\xi}_{rjl}^T = \xi_{rj}^T$.

x_j^{12} the fractional conversion of A achieved in reactions 1 and 2 in temperature interval j.

x_j^{34} the fractional conversion of I achieved in reactions 3 and 4 in temperature interval j.

$x_j^{12^S}$ the fractional conversion of A achieved in reactions 1 and 2 in temperature intervals 1 to j.

$x_j^{34^S}$ the fractional conversion of I achieved in reactions 3 and 4 in temperature intervals 1 to j.

4.8.4 Parameters

$\hat{f}_l^A$ upper bound on $f_A^{R_{in}}$ in feed interval l.

$\hat{f}_l^I$ upper bound on feed of I ($f_I^{R_{in}} + \xi_1$) in feed interval l.

$\hat{t}_{jm}$ time discretization point m for temperature interval j.

T_j Upper bound on temperature in temperature interval j.

$\hat{x}_{jm}^{1^{\max}}$ Maximum fractional conversion achieved in reaction 1 in temperature interval j and time interval m.

$\hat{x}_{jm}^{34^{\min}}$ Minimum fractional conversion achieved in reactions 3 and 4 in temperature interval j and time interval m.

Chapter 5

Siloxane Monomer Case Study

In this chapter screening models are applied to the design of a process for the campaign manufacture of siloxane monomer (Barrera, 1990; Allgor *et al.*, 1996). This example is an abstraction of a problem actually encountered by a major specialty chemical manufacturer. The identities of the compounds involved have been concealed.

The scenario is as follows. Research chemists have recently discovered a new siloxane based polymer, and a significant quantity is now required for test marketing. This example focuses on the development of a campaign to manufacture a fixed quantity of the monomer. Since the development of similar products by competitors is imminent, both the process development activity and the resulting campaign are subject to a strict time horizon constraint. It is also likely that the design will be used to estimate the cost of long term manufacture. Hence, rapid development of an efficient process is pivotal to the success of the new product. The goal of the screening model is to identify favorable process structures quickly, so that these may serve as the starting point for the detailed design.

The process consists of three reaction tasks that manufacture two products; product A is generated in the first reaction and product D is generated in the third. Two applications for the mixed-integer linear screening models are considered. First, the solution from the screening models is compared to that obtained when minimizing the waste generated by the process (Ahmad and Barton, 1995) to examine whether a process generating the minimum amount of waste can make efficient use of equipment and energy. This model contains simple bounds on the extents and selectivity of reaction that can be achieved in the reactors. The second example employs targets for the conversion and selectivity that can be achieved in terms of the operating time and temperature and investigates whether it is cost effective to employ the downstream reaction and separation tasks required to convert intermediate C into product D.

5.1 Laboratory Scale Process

The experimental procedure for the production of siloxane monomer developed by the chemist is a sequential process consisting of batch reaction and distillation tasks. During the bench scale experiments, kinetic expressions governing the reaction mechanisms of the three reaction tasks were developed; these are described in sections 5.1.1 to 5.1.3. In addition, the experiments identified temperature limits required to avoid unwanted side reactions. Both the reaction and distillation tasks must operate below these temperature limits. The batch distillations can operate under vacuum in order to avoid violating these limits. Following Ahmad (1997), we have assumed that pressure changes do not affect the structure of the batch distillation regions. The detailed dynamic models that have been used consider the effect of pressure changes on the performance of the distillation tasks and indicate that the assumption holds.

5.1.1 First Reaction Task

The chemist's experiments determined that the following reaction mechanism best represents the data in the range of temperatures and compositions examined.

$$R1 + R2 \xrightarrow{1} I1 \tag{5.1}$$

$$R1 + I1 \xrightarrow{2} A \tag{5.2}$$

$$I1 \xrightarrow{3} C + H_2 \tag{5.3}$$

$$I1 + C \xrightarrow{4} I2 \tag{5.4}$$

$$I2 \xrightarrow{5} I1 + C \tag{5.5}$$

$$Pt \quad \xrightarrow{6} \quad Pt^* \tag{5.6}$$

Note that the first reaction is catalyzed by the platinum catalyst (Pt); the catalyst can deactivate to Pt^* over the course of the reaction. The chemists discovered that unwanted side reactions are catalyzed at temperatures above 413 K; therefore, such temperatures must be avoided. Further analysis determined that the following expressions best describe the rates of reaction. The constants for these equations are provided in table 5.1 where the units of the preexponential factors (k_{r0}) provide reaction rates in mols^{-1}m^{-3} when the concentrations are measured in mol/m^3.

$$\text{rate}_1 = \kappa_1 C_{R1} C_{R2} \frac{C_{Pt}}{k_7 + C_{Pt}} \tag{5.7}$$

$$\text{rate}_2 = \kappa_2 C_{R1} C_{I1} \tag{5.8}$$

$$\text{rate}_3 = \kappa_3 C_{I1} \tag{5.9}$$

$$\text{rate}_4 = \kappa_4 C_{I1} C_C \tag{5.10}$$

$$\text{rate}_5 = \kappa_5 C_{I2} \tag{5.11}$$

$$\text{rate}_6 = \kappa_6 C_{Pt} \tag{5.12}$$

where the temperature dependence of the rate constants are given by the following Arrhenius expression:

$$\kappa_r = \kappa_{r0} e^{\left\{\frac{-E_a}{RT}\right\}} \qquad\qquad \forall\, r = 1..7$$

5.1.2 Second Reaction Task

The second reaction task converts the intermediate C generated in the first reaction task to a second intermediate E by reacting C with methanol (M) according to the following stoichiometric relationship:

$$M + C \longrightarrow E \tag{5.13}$$

r	E_a [Jmol^{-1}]	κ_{r0}
1	78240	7.50 x 10^4
2	45605	1.01
3	103345	1.22 x 10^{11}
4	32217	3.58 x 10^{-2}
5	91211	7.33 x 10^9
6	0	1.39 x 10^{-4}
7	0	7.00 x 10^{-1}

Table 5.1: Preexponential factors and activation energies defining the rate constants (5.7–5.12) for reactions (5.1–5.6) occurring within the first reaction task

Equation (5.14) defines the rate of reaction (5.13). The chemists imposed an upper temperature limit of 70 K on the operating temperature and determined a rate constant at this temperature of 1.0 m^3/(kmol hr) for concentrations measured in kmol/m^3.

$$\text{rate} = \kappa C_C C_M \tag{5.14}$$

5.1.3 Third Reaction Task

The third reaction task converts the intermediate E generated in the second reaction task to product D by reacting E with water W according to the following stoichiometric relationship:

$$2E + W \longrightarrow D + 2M \tag{5.15}$$

Equation (5.16) defines the rate of reaction (5.13) for the stoichiometry written in (5.15).

$$\text{rate} = \kappa C_C C_M \tag{5.16}$$

The preexponential factor and the activation energy for this reaction are given below:

$$\kappa_0 = 9.142 \times 10^{11} \left[\frac{\mathrm{m}^3}{\mathrm{kmol\ hr}}\right] \tag{5.17}$$

$$E_a = 83354 \left[\frac{\mathrm{J}}{\mathrm{mol}}\right] \tag{5.18}$$

The chemists advise that this reaction is run below 95 C, and this is treated as a design constraint.

5.1.4 Design Constraints

Several design decisions have been made that restrict the operation of the reactors. Total conversion of R2 is required in the first reaction, a minimum of 98% conversion of C to E is required in the second reaction, and a minimum of 85% conversion of E to D is required in the final reaction.

$$f_{1,R2}^{R_{out}} = 0 \tag{5.19}$$

$$(1-.98)\sum_{e\in E} f_{2e}^{R_{in}} \boldsymbol{\rho}_e^T \boldsymbol{\rho}_C \geq f_{2,C}^{R_{out}} \tag{5.20}$$

$$(1-.85)\sum_{e\in E} f_{3e}^{R_{in}} \boldsymbol{\rho}_e^T \boldsymbol{\rho}_E \geq f_{3,E}^{R_{out}} \tag{5.21}$$

Restrictions are also placed on the amount of toluene needed to solvate the first reaction. In addition, an excess of the non-limiting reagent is required in each of the reactions: at least a 15% excess of R1, a three to one ratio of methanol to C, and a 25 to one ratio of water to E are required.

$$\sum_{e\in E} f_{1e}^{R_{in}} \boldsymbol{\rho}_e^T \boldsymbol{\rho}_T \geq 1.5\sum_{e\in E} f_{1e}^{R_{in}} \boldsymbol{\rho}_e^T \boldsymbol{\rho}_{R2} \tag{5.22}$$

$$f_{1,R1}^{R_{out}} \geq .15\sum_{e\in E} f_{1e}^{R_{in}} \boldsymbol{\rho}_e^T \boldsymbol{\rho}_{R1} \tag{5.23}$$

$$\sum_{e\in E} f_{2e}^{R_{in}} \boldsymbol{\rho}_e^T \boldsymbol{\rho}_M \geq 3\sum_{e\in E} f_{2e}^{R_{in}} \boldsymbol{\rho}_e^T \boldsymbol{\rho}_C \tag{5.24}$$

$$\sum_{e\in E} f_{3e}^{R_{in}} \boldsymbol{\rho}_e^T \boldsymbol{\rho}_W \geq 25\sum_{e\in E} f_{3e}^{R_{in}} \boldsymbol{\rho}_e^T \boldsymbol{\rho}_E \tag{5.25}$$

In addition, we require that only toluene, water, methanol, R1, and R2 may be supplied to the process. The product must consist of 98% A and D on a mass basis. Letting $X^{product} = .98$, and $E_P = \{A, D\}$, the purity constraint (3.25) reduces to the following:

$$.98 \sum_e f_e^P \boldsymbol{\rho}_e^T \mathbf{w} \leq f_A^P \boldsymbol{\rho}_A^T \mathbf{w} + f_D^P \boldsymbol{\rho}_D^T \mathbf{w} \tag{5.26}$$

5.2 Case Study I: Comparison of minimum cost versus minimum waste

We require the manufacture of 136,078 kilograms of monomer in less than sixty days. In this problem we compare the difference between minimizing the manufacturing cost and minimizing the manufacturing cost subject to minimum waste emissions. Ahmad (1997) has shown that an embedded optimization that first minimizes the waste emitted by the process and then minimizes the total flow of recycled material while permitting no more than the minimum waste to be emitted leads to sensible process designs with minimum environmental impact. In this section, we compare the difference between minimizing the total cost and minimizing the total cost of a design that emits no more than the minimum amount of waste.

The screening models employed for this case study employ a simplified model of the first reaction task that considers the two dominant reactions given in (5.27–5.28), rather than the set of competing reactions (5.1–5.6). The intermediate species generated in the first reaction are not included in the screening model. We assume that hydrogen remains in the gas phase, and that no cost is incurred when sending the hydrogen to the flare.

$$2R1 + R2 \xrightarrow{1} A \tag{5.27}$$

$$R1 + R2 \xrightarrow{2} C + H_2 \tag{5.28}$$

Toluene is not permitted to mix with water in order to avoid the formation of two liquid phases. We require total conversion of R2 in the first reaction (5.19), so R2 does not appear in the batch distillation regions. Since all of the mixtures in the process are homogeneous, the batch distillation targeting procedure can be employed. Two super simplices are formed, one containing the pure components C, M, R1, W, E, A, and D, and the other containing C, M, R1, T, E, A, and D. The batch distillation regions are extracted from the two super simplices. The batch distillation regions calculated by Ahmad (1997) have been employed; the azeotropic behavior was approximated using the Wilson model to calculate the activity coefficients (see Ahmad (1997) for details). The fourteen distillation regions represented by the product sequences shown in table 5.2 cover the composition space of the allowable distillation feeds. Each super simplex contains seven pure components, so each region is represented by an ordered sequence of seven fixed points taken from the set $E = \{$C, M–T, M, R1–W, R1–T, R1, W–E, W, C–R1–T, C–R1, T, E, A, D$\}$. Since heterogeneous mixtures often appear in specialty chemical process, the separation targets should be extended to include these systems. We recognize that the lower bounds derived from the screening model are subject to the fact that we have imposed the restriction that heterogeneous mixtures are not formed. Note that the mass balances around the distillation tasks forbid the mixing of water and toluene.

Experimental and limited simulation experience has shown that the relative extent that can be achieved in the first reactor at high conversions lies within a restricted range. It should be noted that these bounds are not rigorous, but they serve as suitable bounds for illustration purposes and for a fair comparison with the minimum waste process design found by Ahmad (1997). To compare with the minimum waste solution, we also fix the conversion achieved in the second and third reactions at the lower bounds given by (5.20) and (5.21). These limits are treated as constraints.

$$\xi_{1,1} \geq 1.78\xi_{1,2} \tag{5.29}$$

$$\xi_{1,1} \leq 4.92\xi_{1,2} \tag{5.30}$$

The data needed to implement screening formulation are provided in tables 5.4,

b	Product sequence
1	{C-M, C, R1-W, R1, E, A, D}
2	{C-M, C, R1-W, W-E, W, A, D}
3	{C-M, C, R1-W, W-E, E, A, D}
4	{C-M, M, R1-W, R1, E, A, D}
5	{C-M, M, R1-W, W-E, W, A, D}
6	{C-M, M, R1-W, W-E, E, A, D}
7	{C-M, M-T, M, R2, R1, E, A, D}
8	{C-M, M-T, R1-T, R1, E, A, D}
9	{C-M, M-T, R1-T, T, E, A, D}
10	{C-M, C, C-R1-T, T, E, A, D}
11	{C-M, C, C-R1-T, C-R1, E, A, D}
12	{C-M, R1-T, C-R1-T, C-R1, E, A, D}
13	{C-M, R1-T, C-R1-T, T, E, A, D}
14	{C-M, R1-T, R1, C-R1, E, A, D}

Table 5.2: Feasible product sequences for the first case study of the siloxane monomer process.

5.5, and 5.6. Table 5.3 defines the compositions of the azeotropic fixed points, $\boldsymbol{\rho}_e$; for the pure components, $\boldsymbol{\rho}_e$ is merely a column of the identity matrix and has not been included in the table. The raw material and waste disposal costs for each fixed point are given in table 5.4. The disposal costs are estimates based on the cost for incineration or waste water treatment. Table 5.4 also gives the normal boiling point,

Pure	Fixed Points						
Component	$\boldsymbol{\rho}_{C-M}$	$\boldsymbol{\rho}_{M-T}$	$\boldsymbol{\rho}_{R-W}$	$\boldsymbol{\rho}_{R-T}$	$\boldsymbol{\rho}_{W-E}$	$\boldsymbol{\rho}_{C-R-T}$	$\boldsymbol{\rho}_{C-R}$
C	0.675					0.18	0.31
M	0.325	0.89					
R2							
R1			0.40	0.65		0.30	0.69
W			0.60		0.914		
T		0.11		0.35		0.52	
E					0.086		
A							
D							

Table 5.3: Composition of the fixed points that are not pure components.

the heat of vaporization, and the molar volume and molecular weight of the fixed points. Note that the molar volume and heat of vaporization are underestimates for these quantities over the temperature range that the process operates; the molar volume, molecular weight, and heat of vaporization for the azeotropes represent ideal mixture values. These bounds are chosen so that the ideal mixing rule employed in the screening model bounds the mixture volume and heat of vaporization calculated using an activity coefficient model or equation of state.

e	Raw Material [$ / kg]	Waste Removal [$ / kg]	T_b [K]	H^{vap} [J/mol]	Molar Volume [l / kmol]	Molecular Weight
C-M		16.50	323.4	31250	98.47	138.934
C		16.50	336.6	29300	125.87	190.400
M-T		18.00	337.3	35080	48.89	38.653
M		16.50	337.8	35300	41.56	32.042
R2	8.85	16.50	346.0	29700	128.39	134.320
R1-W		16.50	365.4	40420	38.95	30.841
R1-T		18.00	367.5	37655	83.26	64.801
R1	4.11	16.50	370.0	40000	69.84	50.080
W-E		16.50	370.8	41113	28.50	35.595
W	0.01	1.70	373.2	40700	18.35	18.015
C-R1-T		16.50	373.6	34590	99.87	97.209
I1		16.50	374.0	40300	117.59	192.400
C-R1		16.50	378.8	36683	87.21	93.579
T	1.464	18.00	383.8	33300	108.20	92.141
E		16.50	416.5	45500	136.38	222.430
A		16.50	532.0	62900	131.94	250.480
I2		16.50	618.3	60600	292.44	382.800
D		16.50	752.0	66100	435.50	398.790

Table 5.4: Cost and physical property data for the fixed points.

5.2.1 Solution

We require the manufacture of 300,000 pounds of monomer in less than sixty days. The MILP screening formulation was augmented with the additional constraints (5.19–5.30) and solved using GAMS/OSL (Brooke *et al.*, 1992; IBM, 1991) on an

Reactors		
Volume [gal]	Available Units	Rental Rate [$ / hr]
500	2	50
750	2	70
1250	1	88

Distillation Columns					
Volume [gal]	Vapor Rate [kmol/hr]	Number of Trays	Minimum Reflux Ratio	Available Units	Rental Rate [$ / hr]
750	15	8	1.5	2	90
1000	20	8	1.5	2	110
1250	15	8	1.5	2	125

Table 5.5: Inventory and rental rates for processing equipment.

Utility	Cost [$ / kW yr]
hot	100
cold	25

Table 5.6: Utility cost data for the siloxane monomer example.

HP J200 computer. Two cases have been considered. In the first case, the total production cost was minimized subject to the model constraints. The second case examines the use of an embedded optimization (Ahmad, 1997). In this case, the minimum amount of waste emitted from a process meeting the production requirements was determined first.[1] Next, the manufacturing cost of a process emitting no more than this amount of waste was minimized. The amount of waste that can be emitted is treated as a constraint, and the same objective function (e.g., the manufacturing cost) employed in the first case is used. The solutions of the two cases are compared.

Case IA: Minimum Cost Design

The minimum cost design determined by the screening model chooses to perform two separation tasks and merge the first and second reaction tasks into a single equipment stage. The design employs three reactors and three columns and requires 40 batches to complete the campaign. This design manufactures the product at a cost of $7.40/kg. Figure 5-1 depicts a schematic of the process showing the allocation of equipment and the flow of material in kmols for the campaign. Tables 5.7–5.12 show the breakdown of the costs in the process.

Raw Material Costs				
Raw Material	Cost [$ / kg]	Feed [kg]	Total Cost [$]	$ / kg product
M	1.23	193.03	237.43	0.00
R2	8.85	74104.44	655824.31	4.82
R1	4.11	50769.81	208663.93	1.53
W	0.01	1803.11	18.03	0.00
T	1.46	1525.03	2232.64	0.02
Total		128395.43	866976.35	6.37

Table 5.7: Raw material costs for the entire campaign when minimizing total cost in the first case study.

[1] All waste streams were weighted equally when determining the minimum amount of waste emitted.

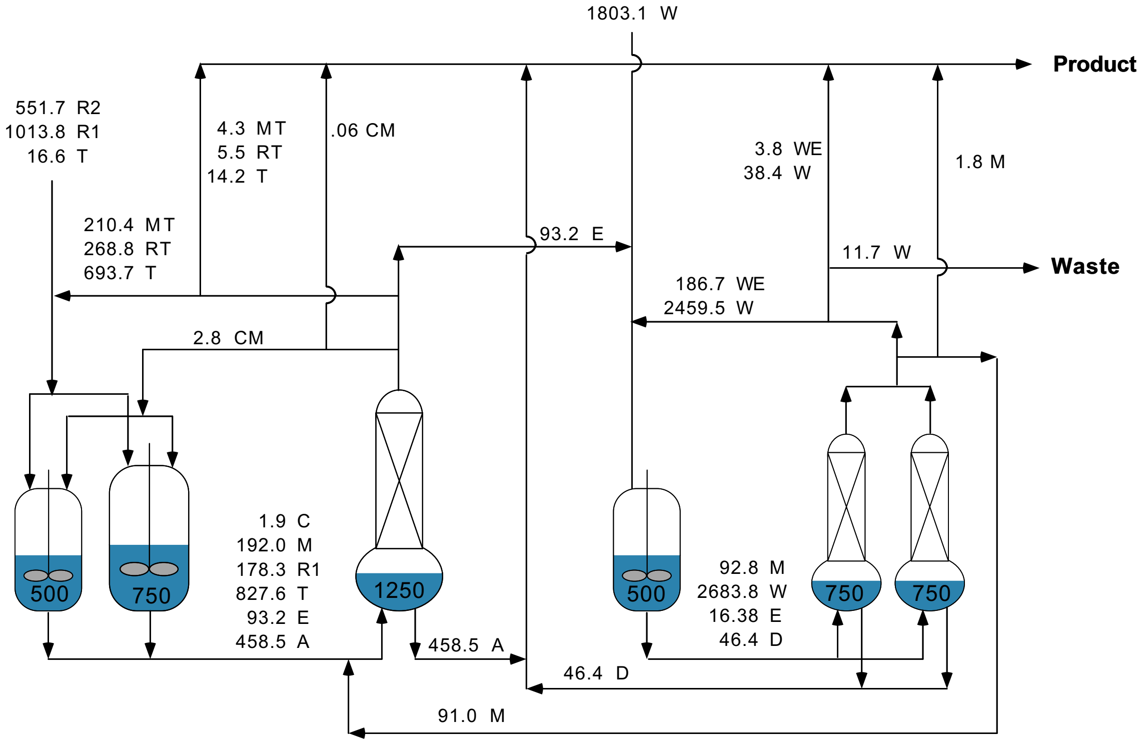

Figure 5-1: Process schematic of the solution derived for Case I.A. Streams labels denote the flow of each fixed point in kmols for the campaign.

Waste Disposal Costs				
Waste Material	Cost [$ / kg]	Amount [kg]	Total Cost [$]	$ / kg product
W	1.70	211.65	359.80	0.00
Total		211.65	359.80	0.00

Table 5.8: Waste disposal costs for the entire campaign when minimizing total cost in the first case study.

Utility Costs			
Cut Material	Amount [kg]	Reboiler Cost [$]	$ / kg product
Distillation 2			
CM	391.30	0.00	0.00
MT	8299.40	0.06	0.00
RT	17772.73	0.08	0.00
T	65230.27	0.19	0.00
E	20720.39	0.03	0.00
Distillation 3			
M	2974.36	0.03	0.00
WE	6780.06	0.06	0.00
W	45212.16	0.81	0.00
Total	167380.68	1.26	0.00

Table 5.9: Utility costs for the entire campaign when minimizing total cost in the first case study.

Reactor Rental Costs					
Volume [gal]		Assigned Units	Rental Rate [$ / hr]	Total Cost [$]	$ per kg product
500		2	50	29275.02	0.22
750		1	70	20492.51	0.15
Distillation Column Rental Costs					
Volume [gal]	Vapor Rate [kmol/hr]	Assigned Units	Rental Rate [$ / hr]	Total Cost [$]	$ per kg product
750	15	2	90	52695.03	0.39
1250	15	1	125	36593.77	0.27
Total for reactors and columns				139056.33	1.02

Table 5.10: Equipment costs for the entire campaign when minimizing total cost in the first case study.

Utilization Measure	Processing Task				
	Rxn 1	Rxn 2	Dist 2	Rxn 3	Dist 3
Cycle Time	1.00	1.00	6.89	1.00	7.32
Volume Required	1192.01	1201.17	1201.17	495.78	495.78
Volume Assigned	1250.00	1250.00	1250.00	500.00	1500.00

Table 5.11: Equipment utilization for the design obtained when minimizing total cost in the first case study.

Cost Contributions			
Component	Percent	Total Cost [$]	$ / kg product
Raw Material	86.15	866976.35	6.37
Waste Disposal	0.04	359.80	0.00
Utility	0.00	1.26	0.00
Equipment	13.82	139056.33	1.02
Total		1006393.73	7.40

Table 5.12: Comparison of raw material, waste disposal, utility, and equipment costs.

Case IB: Minimum Cost subject to Minimum Waste

In this case, a lower bound on the mass of waste emitted by a process meeting the production requirements was determined by minimizing the following objective function:

$$\phi^{\text{waste}} = \sum_{e \in E} f_e^{Waste} w_e \tag{5.31}$$

subject to the mass balance constraints in the screening model (i.e., (3.5–3.7, 3.11, 3.13, 3.14–3.23, and 3.24–3.26)) and the design and reaction targeting constraints presented in this chapter. The solution of the resulting MILP determined that at least 211.65 kg of waste must be emitted from a process meeting the production requirement.[2] Next, a design with minimum cost that does not emit more than this much waste was determined by adding the following constraint to the model solved in Case IA:

$$\sum_{e \in E} f_e^{Waste} w_e \leq \phi^{\text{waste}} \tag{5.32}$$

Since only 211.65 kg of waste is emitted by the solution of Case IA, (5.32) is satisfied by the solution of Case IA, and the solution to this problem is the same as the solution to Case IA. For this example, the solution with minimum environmental impact (measured by the total mass of waste emitted) is the same as the solution with minimum cost.

Next, we examine how the structure of the process defined by the solution to this problem compares to structure of the minimum waste process found by Ahmad (1997). In her method, first the minimum amount of waste is determined, and then the total flow of recycled material is minimized subject to the minimum waste constraint. In this method, the first minimization is the same as the first subproblem solved in Case IB, except that she minimized the total moles of waste $\sum_{e \in E} f_e^{Waste}$ rather than the mass. The second subproblem that she solves differs from the second problem solved here because the procedure used by Ahmad (1997) does not consider the equipment costs. Instead, she minimizes the total flow of recycled material. We compare these results to see whether considering the equipment costs during the optimization changes the structure of the resulting process for this example.

Surprisingly, the design obtained from the solution of Case 2 has the same process structure as the design found by Ahmad (1997), in which total flow of recycled material was minimized subject to the minimum emission requirement. Although equipment costs were not considered in the approach taken by Ahmad (1997), less waste is generated by eliminating the first distillation task, so the processing structure happens to be the same. Section 5.3 shows that this occurs because the methanol in-

[2] Note that the waste generated is small compared to the 136,078 kg of product that is manufactured.

troduced in the second reaction task avoids generating C-$R1$-T and C-$R1$ azeotropes. However, if the minimum emission is specified on a molar basis (as in Ahmad (1997)), the solution of case IA does not satisfy the minimum waste requirement (even though the operation of the distillation and reaction tasks is the same). Both designs emit 211.65 kg of waste, yet the minimum cost design emits water (which costs less, but contains more mols), and the solution of IB emits toluene and the C-M azeotrope because fewer moles are contained in the same mass of waste. These results demonstrate that for some problems in which the material and waste costs dominate, the embedded optimization procedure presented by Ahmad (1997) may generate a process structure leading to a favorable design from a cost point of view.

We have ignored the end effects during the design of these processes, yet the recycled material will need to be disposed at the end of the campaign. In these designs, the amount of material recycled per batch is known, and this provides a good estimate for the amount of waste that may be generated at the end of the campaign. Since 2% (one fiftieth) of the recycled material is purged, but only 40 batches are required, the amount of waste generated by disposing of the recycled material at the conclusion of the campaign is greater than the amount purged during the duration of the campaign, if the the design is not modified to account for the cost of this waste disposal. If we assume that we must simply dispose of this material (i.e., no change in the operation of the process near the end of the campaign is considered) then we can incorporate this cost into our objective function. It may be advantageous to employ a greater number of smaller batches during the campaign, balancing equipment and waste disposal costs. This is investigated in section 5.3.3; we employ the reaction targeting model explained in the next section in order to consider the reaction time, which impacts the tradeoff between the number of batches employed and the length of the campaign.

5.3 Case Study II: Including Reaction Targets

This example demonstrates that bounds can be derived for the reaction tasks in this process. In this example, we consider partial conversion of R2, and we account for the intermediate components I1, and I2. These components do not form azeotropes with any of the other components in the system.[3] Table 5.13 shows the distillation regions for this process.

b	Product sequence
1	{C-M, C, R2, R1-W, R1, I1, E, A, I2, D}
2	{C-M, C, R2, R1-W, W-E, W, I1, A, I2, D}
3	{C-M, C, R2, R1-W, W-E, I1, E, A, I2, D}
4	{C-M, M, R2, R1-W, R1, I1, E, A, I2, D}
5	{C-M, M, R2, R1-W, W-E, W, I1, A, I2, D}
6	{C-M, M, R2, R1-W, W-E, I1, E, A, I2, D}
7	{C-M, M-T, M, R2, R1, I1, E, A, I2, D}
8	{C-M, M-T, R2, R1-T, R1, I1, E, A, I2, D}
9	{C-M, M-T, R2, R1-T, I1, T, E, A, I2, D}
10	{C-M, C, R2, C-R1-T, I1, T, E, A, I2, D}
11	{C-M, C, R2, C-R1-T, I1, C-R1, E, A, I2, D}
12	{C-M, R2, R1-T, C-R1-T, I1, C-R1, E, A, I2, D}
13	{C-M, R2, R1-T, C-R1-T, I1, T, E, A, I2, D}
14	{C-M, R2, R1-T, R1, I1, C-R1, E, A, I2, D}

Table 5.13: Feasible product sequences for the second case study of the siloxane monomer process.

5.3.1 First Reaction Task

Targets have been developed for the reaction tasks. These targets consider all of the components in the reactions, except for the catalyst. We ignore the limitation on the reaction rate imposed by the deactivation of the catalyst, so only five reactions

[3] The property estimation methods indicate that R2 does not behave ideally, but the predicted interactions were not realistic, so for the purposes of illustration R2 has been assumed to interact ideally. Note that for the design of an industrial process, experimental VLE data defining the interaction of R2 would be crucial to the validity of the results.

(5.1–5.5) are considered in this case study. This assumption maintains the bounding property of the screening model.

Upper bounds on the extent of reaction in terms of the operating time and temperature are enforced on all of the reactions except for the reversible reaction (5.4–5.5). Since (5.4–5.5) denote a reversible reaction, the extents of these reactions can be unbounded since the mass balance is satisfied if any feasible values for ξ_{14} and ξ_{15} are both increased by an arbitrary constant. The difference between ξ_{14} and ξ_{15} is the quantity with which we are concerned. We bound the ξ_{15} according to the amount of I2 charged to provide a reference for the extent of these reactions.

$$\xi_{15} \leq f_{I2}^{R_{in}} \tag{5.33}$$

Given the reference established by (5.33), all of the extents are bounded by the mass balances. In addition, we place bounds on the extents of the first three reactions in terms of the reaction time, temperature, and the amount of material charged to the task. For the first and second order reactions occurring in this task, the conversion of material per unit volume will always be less than the conversion that would be achieved if the same material occupied a smaller volume. Thus, the following upper bounds can be placed on the extents of the first three reactions:

$$\frac{d\xi_{11}}{dt} \leq \kappa_{11}(T)\frac{N_{R1}N_{R2}}{V^{\min}} \leq \kappa_{11}(T)C_{R1}^{\max}N_{R2} \tag{5.34}$$

$$\frac{d\xi_{12}}{dt} \leq \kappa_{12}(T)\frac{N_{R1}N_{I1}}{V^{\min}} \leq \kappa_{12}(T)C_{R1}^{\max}N_{I1} \tag{5.35}$$

$$\frac{d\xi_{12}}{dt} = \kappa_{13}(T)N_{I1} \tag{5.36}$$

where $C_{R1}^{\max}$ represents a rigorous upper bound on the concentration of R1 in the reactor. The maximum extents of reaction can be achieved when operating at the maximum temperature and when all of the reactants are available at the initial time. Upper bounds on ξ_{11}, ξ_{12}, and ξ_{13} are derived by assuming that the maximum rates given by the expressions above can be achieved and solving (5.34–5.36). Since we bound the selectivity according to the temperature at which the reactions occur, the feasible operating temperature range is divided into intervals following the procedure employed in chapter 4. Since $R2$ can be converted to $I1$ at one temperature and converted to either A or C at another temperature, we cannot assume that the only $I1$ available at the start of any temperature interval is that which is charged directly to the reactor. We make the assumption that all of the $I1$ generated by reaction 1 is available at the initial time, which preserves the bounding property of the model. Thus, to bound the extents of reaction, (5.34–5.36) is solved for the initial conditions $\xi_{11}(0) = \xi_{12}(0) = \xi_{13}(0) = 0$, $N_{R2}(0) = N^o_{R2}$, $N_{I1}(0) = N^o_{I1} + \xi_{11}$, which leads to the following upper bounds on the extents of reaction:

$$\xi_{11}(t^R_1) \leq N^o_{R2}\left(1 - e^{-\alpha_1 t^R_1}\right) \tag{5.37}$$

$$\xi_{12}(t^R_1) \leq \left(N^o_{I1} + \xi_{11}\right)\frac{\alpha_2}{\alpha_3}\left(1 - e^{-\alpha_3 t^R_1}\right) \tag{5.38}$$

$$\xi_{13}(t^R_1) \leq \left(N^o_{I1} + \xi_{11}\right)\frac{\kappa_{13}(T^{\max})}{\alpha_3}\left(1 - e^{-\alpha_3 t^R_1}\right) \tag{5.39}$$

where

$$\alpha_1 = \kappa_{11}(T^{\max})C^{\max}_{R1} \tag{5.40}$$

$$\alpha_2 = \kappa_{12}(T^{\max})C^{\max}_{R1} \tag{5.41}$$

$$\alpha_3 = \kappa_{13}(T^{\max}) + \alpha_2 \tag{5.42}$$

An upper bound on the the extents of the competing reactions can be expressed as follows:

$$\xi_{12} + \xi_{13} \leq \left(N^o_{I1} + \xi_{11}\right)\left(1 - e^{-\alpha_3 t^R_1}\right) \tag{5.43}$$

The bounds on the extent of reaction depend on the charge of material and a function of the temperature, concentration of R1, and the time. Following the procedure employed in chapter 4, these bounds on the extents are expressed in terms of the new variables x^1 and x^{23} that account for the time, temperature, and concentration dependence:

$$\xi_{11} \leq N^o_{R2} x^1 \tag{5.44}$$

$$\xi_{12} + \xi_{13} \leq \left(N^o_{I1} + \xi_{11}\right) x^{23} \tag{5.45}$$

$$x^1 = 1 - e^{-\alpha_1 t^R_1} \tag{5.46}$$

$$x^{23} = 1 - e^{-\alpha_3 t^R_1} \tag{5.47}$$

As shown in chapter 4 these bilinear expressions do not define a convex feasible region. However, for given values of $T^{\max}$ and $C^{\max}_{R1}$ the hypograph of the functions x^1 and x^{23} define convex regions. Overestimates for the variables x^1 and x^{23} are derived as follows. First, the feasible temperature range is partitioned into a set of temperature intervals, denoted by the subscript j, so that $T^{\min} = T_0 < \ldots T_j < T^{\max}$. Next, a bound on the maximum concentration of R1 in the reactor is defined in terms of the ratio of R1 to R2 fed to the reactor. The maximum of concentration of R1 in the reactor is partitioned into intervals denoted by the subscript c. In each of these intervals, γ_c defines the upper limit of the ratio of R1 to R2 and $\hat{C}_c^{R1^{\max}}$ defines an upper bound on the maximum concentration of R1 that is possible. An integer variable $y_c^{C^{\max}_{R1}}$ is used to indicate the overall ratio of R1 to R2 charged as follows:

$$\gamma_{c-1} y_c^{C^{\max}_{R1}} f^{R_{in}}_{kR2} \leq y_c^{C^{\max}_{R1}} \sum_e f^{R_{in}}_{ek} \rho_{e_{R1}} \qquad \forall\, c \geq 1, k = 1 \tag{5.48}$$

$$y_c^{C^{\max}_{R1}} \sum_e f^{R_{in}}_{ek} \rho_{e_{R1}} \leq \gamma_c y_c^{C^{\max}_{R1}} f^{R_{in}}_{kR2} \qquad \forall\, c, k = 1 \tag{5.49}$$

$$\sum_c y_c^{C^{\max}} = 1 \tag{5.50}$$

A large value for γ_{n_c} was selected so that these equations can always be satisfied for some value of $y_c^{C^{\max}_{R1}}$, but the maximum concentration of $R1$ in the last interval (i.e., $c = n_c$) is defined by the molar volume of $R1$ (see (5.52)), knowing that the concentration can never be higher than that of the pure component. The maximum concentration of R1 in each of these intervals is defined from the knowledge that the number of moles of toluene charged to the reactor must be at least 1.5 times the amount of R2 charged. Since the solvent toluene is required to be in the reactor during the entire reaction, the maximum concentration of R1 can be determined assuming that only toluene and R1 are present. Thus, an upper bound on the maximum concentration of R1 in each of the c intervals can be calculated as follows:

$$\hat{C}_c^{R1^{\max}} = \frac{\gamma_c}{\gamma_c v_{R1} + 1.5 v_T} \qquad \text{if } c < n_c \tag{5.51}$$

$$\hat{C}_c^{R1^{\max}} = \frac{1}{v_{R1}} \qquad \text{if } c = n_c \tag{5.52}$$

We define values $\hat{\alpha}_{1_{cj}}$–$\hat{\alpha}_{4_{cj}}$ corresponding to $T_j = T^{\max}$ and $\hat{C}_c^{R1^{\max}}$ that overestimate the rates of reaction when operating in temperature interval j and concentration interval c. We assign the variable t_j^T to denote the time the first reaction spends in temperature interval j.[4] The variables x_{cj}^1 and x_{cj}^{23} are defined in terms of these parameters and t_j^T as follows:

$$x_{cj}^1 = 1 - e^{-\hat{\alpha}_{1_{cj}} t_j^T} \qquad \forall\ c, j \tag{5.53}$$

$$x_{cj}^{23} = 1 - e^{-\hat{\alpha}_{3_{cj}} t_j^T} \qquad \forall\ c, j \tag{5.54}$$

Since x_{cj}^1 and x_{cj}^{23} are defined by concave functions, tangents to these functions overestimate the feasible region that defines the reaction extents in terms of time. We pick m discrete points in time ($\hat{t}_{cjm}$) for each temperature and concentration interval at which we define the tangents to the function. The region lying beneath the tangent curves overestimates the hypograph of x_{cj}^1 and x_{cj}^{23}. These tangents generate the following bounds:

$$x_{cj}^1 \leq 1 - e^{-\hat{\alpha}_{1_{cj}} \hat{t}_{cjm}} + \hat{\alpha}_{1_{cj}} e^{-\hat{\alpha}_{1_{cj}} \hat{t}_{cjm}} \left(t_j^T - \hat{t}_{cjm}\right) \qquad \forall\ c, j, m \tag{5.55}$$

$$x_{cj}^{23} \leq 1 - e^{-\hat{\alpha}_{3_{cj}} \hat{t}_{cjm}} + \hat{\alpha}_{3_{cj}} e^{-\hat{\alpha}_{3_{cj}} \hat{t}_{cjm}} \left(t_j^T - \hat{t}_{cjm}\right) \qquad \forall\ c, j, m \tag{5.56}$$

Bounds on the extents of reaction in each temperature interval are calculated by

[4] Note that the performance of the other reactions is not assigned to different temperature intervals, so t_j^T applies only to the first reaction.

employing linear overestimators (4.36–4.37) (McCormick, 1976) for the bilinear expressions in (5.44) and (5.45). Following the same procedure used in chapter 4, the total charge of N^o_{R2} and $N^o_{I1} + \xi_{11}$ are divided into intervals denoted by the subscript l, and the active feed intervals for R2 and I1 are identified by a binary variables $y^{F_{R2}}_l$ and $y^{F_{I1}}_l$:

$$\sum_{l \in L} y^{F_{R2}}_l = 1 \tag{5.57}$$

$$\sum_{l \in L} y^{F_{I1}}_l = 1 \tag{5.58}$$

$$\sum_{l \in L} \hat{f}^{R2}_{l-1} y^{F_{R2}}_l \leq f^{R_{in}}_{R2} \leq \sum_{l \in L} \hat{f}^{R2}_{l} y^{F_{R2}}_l \tag{5.59}$$

$$\sum_{l \in L} \hat{f}^{I1}_{l-1} y^{F_{I1}}_l \leq f^{R_{in}}_{I1} + \xi_1 \leq \sum_{l \in L} \hat{f}^{I1}_{l} y^{F_{I1}}_l \tag{5.60}$$

To employ this strategy the variables $\tilde{\xi}^T_{1cjl} = \xi^T_{1cj} y^{F_{R2}}_l$ are introduced to denote the extent of reaction 1 in concentration interval c, temperature interval j, and feed interval l, so $\sum_c \sum_j \sum_l \tilde{\xi}^T_{1cjl} = \xi_{11}$. In addition, the variables $\tilde{x}^1_{1cjl} = x^1_j y^{F_{R2}}_l$, $\tilde{f}^{R_{in}}_{1cR2l} = y^{F_{R2}}_l f^{R_{in}}_{R2}$, and $\tilde{f}^{R_{in}}_{I1l} = y^{F_A}_l (f^{R_{in}}_{I} + \xi_{11})$ are introduced. The same procedure is applied for reactions 2 and 3. Note that $\sum_{l \in L} \tilde{\xi}^T_{1rjl} = \xi^T_{1rj} \quad \forall\ j, r = 1..3$.

$$\tilde{\xi}^T_{1cjl} \leq \hat{f}^{R2}_{l-1} \tilde{x}^1_{cjl} - \hat{f}^{R2}_{l-1} y^{F_{R2}}_l + \tilde{f}^{R_{in}}_{1R2l} \qquad \forall\ c, j \in J, l \in L \tag{5.61}$$

$$\tilde{\xi}^T_{1cjl} \leq \hat{f}^{R2}_{l} x^1_{cj} \qquad \forall\ c, j \in J, l \in L \tag{5.62}$$

and similar constraints are derived to bound the extents of reactions 2 and 3:

$$\tilde{\xi}^T_{2cjl} + \tilde{\xi}^T_{3cjl} \leq \hat{f}^{I1}_{l-1} \tilde{x}^{23}_{cjl} - \hat{f}^{I1}_{l-1} y^{F_{I1}}_l + \tilde{f}^{R_{in}}_{1I1l} \qquad \forall\ c, j \in J, l \in L \tag{5.63}$$

$$\tilde{\xi}^T_{2cjl} + \tilde{\xi}^T_{3cjl} \leq \hat{f}^{I1}_{l} x^{23}_{cj} \qquad \forall\ c, j \in J, l \in L \tag{5.64}$$

Constraints to bound the extents in consecutive temperature intervals analogous to (4.52–4.55) are also derived and included with the screening model.

An upper bound on the selectivity of reaction 2 to 3 is imposed in each temperature interval based on the relative rates of reaction. The ratio of the rate of reaction 2 to 3 is defined as follows:

$$\frac{\mathrm{rate}_2}{\mathrm{rate}_3} = \frac{\kappa_{12}(T) C_{R1} C_{I1}}{\kappa_{13}(T) C_{I1}} = \frac{\kappa_{12}(T)}{\kappa_{13}(T)} C_{R1} \tag{5.65}$$

Since the selectivity is a function of only temperature and the concentration of $R1$ and the activation energy of reaction 2 is less than that of reaction 3, the selectivity can be bounded in each temperature interval j as follows:

$$\frac{\mathrm{rate}_2}{\mathrm{rate}_3} \leq \frac{\kappa_{12}(T_{j-1})}{\kappa_{13}(T_{j-1})} C_{R1}^{\max} \tag{5.66}$$

Since $C_{R1}^{\max}$ is bounded by the active feed interval, (5.66) can be expressed as follows:

$$\sum_l \tilde{\xi}_{2cjl}^T \leq \frac{\kappa_{12}(T_{j-1}) \hat{C}_c^{R1^{\max}}}{\kappa_{13}(T_{j-1})} \sum_l \tilde{\xi}_{3cjl}^T \quad \forall\ c, j \in J \tag{5.67}$$

These bounds assume that the concentration of $R1$ is held constant throughout the reaction, so they are rigorous but may not be very tight. Since the selectivity varies exponentially with temperature and only linearly with concentration, these bounds capture the dominating tradeoff.

Second Reaction Task

The following reaction occurs in the second reaction task:

$$C + M \longrightarrow E \tag{5.68}$$

In the second reaction task an upper bound on the reaction rate can be obtained by by overestimating the concentration of methanol in the reactor.

$$\frac{d\xi_{21}}{dt} = \kappa_{21} \frac{N_C N_M}{V} \leq \kappa_{21} \frac{N_C N_M}{V_{\min}} \leq \kappa_{21} N_C C_M^{\max} \leq \frac{\kappa_{21} N_C}{V_M} \tag{5.69}$$

We have employed the fact that the concentration of methanol cannot be greater than the concentration of pure methanol, defined by the molar volume of the species. While this is a crude approximation, it is not too far (within 30 %) from the initial methanol concentration if there are no other solvents in the reactor, and it provides a rigorous bound. Therefore,

$$\xi_{21} \leq f_{2C}^{R_{in}} \left(1 - e^{-\frac{\kappa_{21}}{v_M} t_2^R}\right) \tag{5.70}$$

Since the second reaction task did not typically limit the cycle time, this bound was deemed sufficient.

For this reaction task, linear bounds are enforced by providing a piecewise constant overestimation of the feasible region. Since high conversion is required in this reactor, the conversion is divided into intervals, and an SOS1 set (i.e., $\sum_c y_c^{\text{Conv}^C} = 1$) of binary variables $y_c^{\text{Conv}^C}$ is employed to indicate in what range the conversion achieved lies. Denoting the upper and lower bounds on the conversion in each interval by $\hat{x}_c^{C^{UP}}$ and $\hat{x}_C^{C^{LO}}$ respectively, the active region can be identified as follows:

$$\sum_c y_c^{\text{Conv}^C} \hat{x}_c^{C^{LO}} f_{2C}^{R_{in}} \leq \xi_{21} \leq \sum_c y_c^{\text{Conv}^C} \hat{x}_c^{C^{UP}} f_{2C}^{R_{in}} \tag{5.71}$$

The bilinear terms are replaced by introducing a new variable $N_c^C = f_{2C}^{R_{in}} y_c^{\text{Conv}^C}$ defined using an exact linearization. A lower bound on the time required to achieve ξ_{21} is given as follows:

$$t_2^R \geq \sum_c y_c^{\text{Conv}^C} \frac{\ln\left(1 - \hat{x}_c^{C^{LO}}\right)}{-\kappa_{21}/V_M} \tag{5.72}$$

Third Reaction Task

The third reaction task converts intermediate E into product D. The reaction is carried out in a large excess of water (at least 25 times E). This reaction is restricted to temperatures below 95 C, so an upper bound on the rates that can be achieved is imposed by this temperature. Since the reaction is second order in E, the rates will be maximized if the same material is contained in a smaller volume. This implies that the rate can be overestimated by assuming no dilution by inert materials, by underestimating the volume during the entire reaction, and by assuming that all the reactants are available initially. The volume increase upon reaction is ignored to overestimate the rate of reaction. If the reaction is carried out isothermally, the reaction time can be related to the conversion of E as follows:

$$2\kappa_{31}t_3^R = \frac{1}{C_E} - \frac{1}{C_{E_o}} = \frac{V_f}{(1-x^E)N_{E_o}} - \frac{V_o}{N_{E_o}} \geq \frac{V_o}{N_{E_o}}\left(\frac{1}{1-x^E} - 1\right) \tag{5.73}$$

An underestimate of the time required to achieve a given conversion occurs can be derived from (5.73) by assuming the reactor is operated at the maximum temperature for the duration of the reaction and by underestimating the V_o/N_{E_o} term by assuming the concentration of E is not diluted by excess water or other components.

$$t_3^R \geq \left(\frac{1}{1-x^E} - 1\right)\left(\frac{V_E + 25V_W}{2\kappa_{31}^{\max}}\right) \tag{5.74}$$

In (5.74) $\kappa_{31}^{\max}$ denotes the value of the rate constant at 95 C.

Equation (5.74) can be used to derive a simple lower bound on the reaction time in the same fashion used to derive a lower bound for the processing time of the second reaction. The conversion that is achieved in the reactor is restricted to lie in one of several intervals that cover the range of feasible conversions for this reaction; a SOS1 set of binary variables $y_c^{\text{Conv}^E}$ (i.e., $\sum_c y_c^{\text{Conv}^E} = 1$) is employed to indicate in what range the conversion achieved lies. Denoting the upper and lower bounds on the conversion in each interval by $\hat{x}_c^{E^{UP}}$ and $\hat{x}_c^{E^{LO}}$ respectively. New variables $N_c^E = f_{3E}^{R_{in}} y_c^{\text{Conv}^C}$ are defined using an exact linearization in order to relate the conversion to the extent of the reaction.

$$\sum_c N_c^E \hat{x}_c^{E^{LO}} \leq 2\xi_{31} \leq \sum_c N_c^E \hat{x}_c^{E^{UP}} \tag{5.75}$$

A lower bound on the time required to achieve ξ_{31} is given by replacing the x^E in

(5.74) with the lower bound of the active conversion interval:

$$t_3^R \geq \left(\frac{V_E + 25V_W}{2\kappa_3^{\max}}\right) \sum_c y_c^{\mathrm{Conv}^E} \left(\frac{1}{1 - \hat{x}_c^{E^{LO}}} - 1\right) \tag{5.76}$$

5.3.2 Solutions to Case Study II

The screening model was employed to determine the minimum cost design for the production of siloxane monomer. In determining the minimum cost design the screening model determines whether the downstream processing to convert C into D is cost efficient. Two superstructures were considered in this case study. The first includes only the first reaction task, and the second requires all three. The screening model will select between these two options if it is allowed to decide whether the reaction tasks should be performed or not, but solving the problem using two different superstructures allows us to compare the optimal screening solution derived from each superstructure, rather than simply finding out which structure leads to the best solution. In addition, it reduces the combinatorial complexity of the model.

Lower bounds on the manufacturing cost for 136,078 kg of product are determined for each superstructure. Raw material, waste disposal, utility, and equipment rental costs were considered for a manufacturing campaign employing no intermediate storage; end effects were ignored. The product was required at a purity of 98% defined on a mass basis. Two percent of all recycled material was purged. Material transfers are assumed to take .5 hours, and .5 hours are required to bring the columns to total reflux before drawing product. The solutions obtained for each of the superstructures are described in the next two sections.

Eight temperature intervals defined by (310, 320, 330, 340, 350, 360, 370, 390, 410 K) were employed when deriving the targets for the first reaction task. Five feed intervals were employed. The upper bound on the first four feed intervals represent increases of 2 % over the minimum amount of $R2$ required to achieve the required production. The ratio of $R1$ to $R2$ was partitioned into five intervals, with the upper bounds on the first four intervals defined by 2.0, 2.5, 3.0, and 4.0.

Case II.A: One Reaction Task Allowed

A lower bound on the manufacturing cost of $6.59/kg was obtained using only one reaction and one distillation task. A schematic of the solution is provided in figure 5-2. The streams are labeled with the material flows for the entire campaign for each of the fixed points contained in the stream. Since 45 batches are employed in this campaign, the amounts charged during each batch can be determined from the figure 5-2.

The solution of the screening model for the three reaction process, chooses not to perform any of the second and third reactions even though we imposed constraints that required equipment to be assigned to the third reaction and distillation tasks. Hence, it cost more that the solution above.

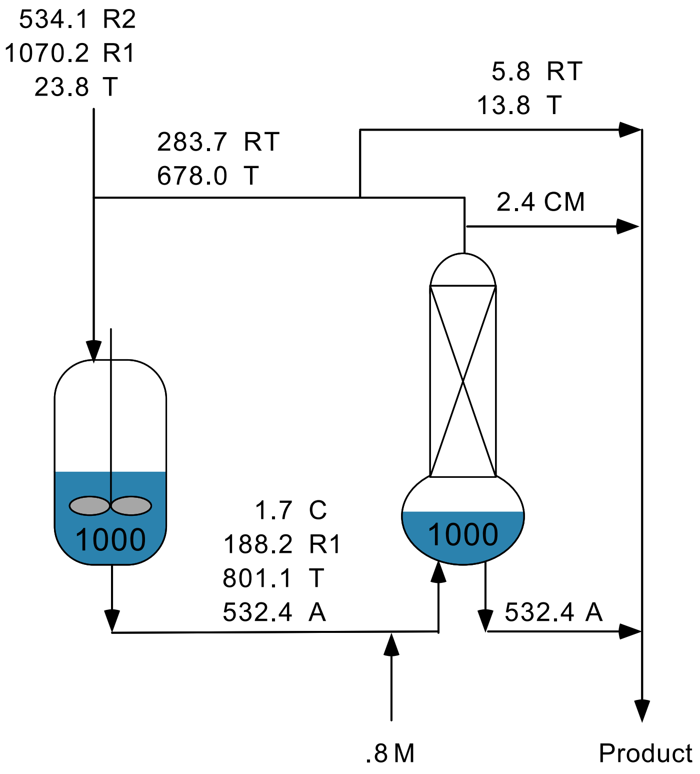

Figure 5-2: Process schematic of the solution derived from the superstructure containing only one reaction task. Streams labels denote the flow of each fixed point in kmols for the campaign.

The reaction task employs the 1000 gallon reactor and converts all of the $R2$ charged into A and C, with no $I1$ left unreacted. The reactor operates for a total of 3.25 hours, spending over 99 % of the processing time in the first temperature interval. Seventy percent of the extent of the first reaction is achieved in the first temperature interval, and over 96 % of the extent of the second and third reactions can be attributed to the time spent in the first temperature interval. The other 30% of the extent of the first reaction is attributed to the time spent in the higher temperature intervals, with 20% being attributed to time spent in the 370-410 K range. A high selectivity is achieved by operating at a low temperature. The performance given by the screening model represents a bound on the performance of an actual reactor, so the detailed dynamic model of the reaction task may not be able to achieve the performance predicted by the screening model.

In order to operate at cyclic steady state, any C generated by the reaction task must be removed from the process. It can be removed as either impurity in the product or as waste; the screening model generates no waste by incorporating all of the C generated in the process as impurity in the product. In order to remove this C at minimum cost, the screening model chooses to add methanol to the feed to the distillation column. This permits the C to be removed in the C-M azeotrope and prevents the formation of the C-$R1$-T azeotrope. Although the screening model does not consider the difficulty of the separation task (e.g., the purity of each cut employed in the detailed process design and whether the fixed points are close boiling), the use of methanol as an entrainer makes the separation of C from the rest of the components easier, because C is removed in the minimum boiling azeotrope formed between C and methanol that has a normal boiling point below all of the other fixed points in the system. The reactor effluent combined with the methanol places the feed to the distillation task in batch distillation region 9. The first overhead cut contains the C-M azeotrope, which is sent to waste. The next overhead cut contains the solvent and unused reagents and is recycled to the reaction task. The product A is taken in the bottoms of the column.

Raw material costs dominate the production costs for this design, as shown in table 5.18. Tables 5.14, and 5.15 show the material processing costs for the campaign. Table 5.16 shows the charges incurred for the use of equipment during the campaign, and table 5.16 shows the utilization of the equipment. The batch size and cycle time are the same for both tasks.

Raw Material Costs				
Raw Material	Cost [$ / kg]	Feed [kg]	Total Cost [$]	$ / kg product
M	1.23	25.48	31.34	0.00
R2	8.85	71734.25	634848.15	4.67
R1	4.11	53596.72	220282.52	1.62
T	1.46	2193.08	3210.67	0.02
Total		127549.54	858372.68	6.31

Table 5.14: Raw material costs for the entire campaign for the process containing only one reaction task.

Utility Costs			
Cut Material	Amount [kg]	Reboiler Cost [$]	$ / kg product
Distillation 1			
CM	339.98	0.00	0.00
RT	18762.33	0.09	0.00
T	64475.18	0.18	0.00
Total	83577.48	0.27	0.00

Table 5.15: Utility costs for the distillations for the entire campaign for the process containing only one reaction task.

Case II.B: All Reaction Tasks Required

A lower bound on the manufacturing cost of $6.80/kg is obtained when all we require that all three reaction tasks are performed. In order to ensure that all the reactions are performed, we require that at least 98% of the C generated in the first reaction is converted to E in the in the second reaction, and we require that at least 85% of the E is converted into D. A schematic of the solution is provided in figure 5-2. The streams are labeled with the material flows for the entire campaign for each of the fixed points contained in the stream. Since 36 batches are employed in this campaign, the amounts charged during each batch can be determined from the figure 5-3.

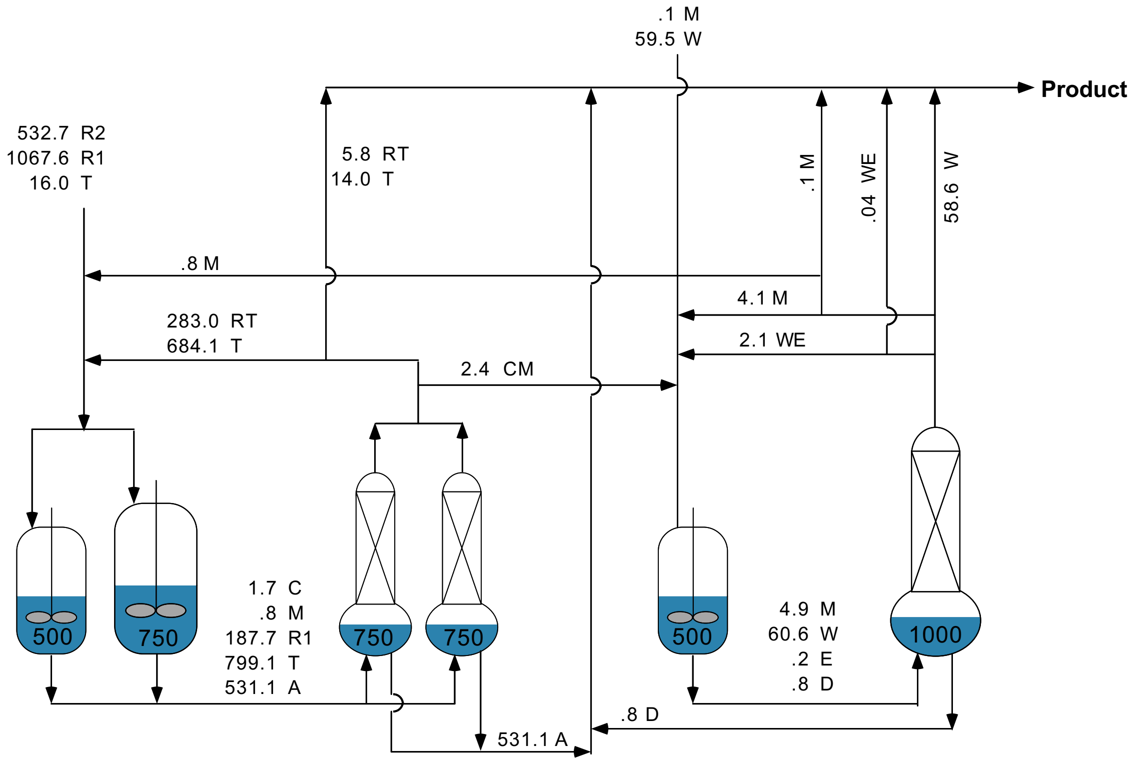

Figure 5-3: Process schematic of the solution derived from the superstructure requiring all three reaction tasks. Streams labels denote the flow of each fixed point in kmols for the campaign.

Reactor Rental Costs					
Volume [gal]		Assigned Units	Rental Rate [$ / hr]	Total Cost [$]	$ per kg product
1000		1	88	16849.01	0.12
Distillation Column Rental Costs					
Volume [gal]	Vapor Rate [kmol/hr]	Assigned Units	Rental Rate [$ / hr]	Total Cost [$]	$ per kg product
1000	20	1	110	21061.26	0.16
Total for reactors and columns				37910.26	0.28

Table 5.16: Equipment costs for the entire campaign for the process containing only one reaction task.

Utilization Measure	Processing Task	
	Reaction 1	Distillation 1
Cycle Time	4.25	4.25
Volume Required	999.58	999.78
Volume Assigned	1000.00	1000.00

Table 5.17: Equipment utilization for the design obtained from the process containing one reaction task.

Cost Contributions			
Component	Percent	Total Cost [$]	$ / kg product
Raw Material	95.77	858372.68	6.31
Waste Disposal	0.00	0.00	0.00
Utility	0.00	0.27	0.00
Equipment	4.23	37910.26	0.28
Total		896283.22	6.59

Table 5.18: Comparison of raw material, waste disposal, utility, and equipment costs for the process containing only one reaction task.

The first reaction task employs the 500 and 750 gallon reactors and converts all of the $R2$ charged into A and C, with no $I1$ left unreacted. The reactor operates for a total of 3.27 hours, spending over 99 % of the processing time in the first temperature interval. Roughly seventy percent of the extent of the first reaction is achieved in the first temperature interval, and over 96 % of the extent of the second and third reactions can be attributed to the time spent in the first temperature interval. The other 30% of the extent of the first reaction is attributed to the time spent in the higher temperature intervals. A high selectivity is achieved by operating at a low temperature. The operation of the first reactor given by the solution is very similar to the reactor operation for Case II.A. However, in this case, we require that the C generated in the first reactor is processed to product D.

The effluent from the first reactor is separated using both 750 gallon distillation columns. The columns operate in batch distillation region 9; note that the methanol recycled from the third distillation to the first reactor acts as an entrainer. A is taken in the bottoms of the column, the C-M azeotrope is passed on to the second reaction, and the unused reagent and solvent are recycled.

The second and third reaction tasks are merged into a single equipment stage that employs a 500 gallon reactor. The effluent from these reaction tasks is separated in the 1000 gallon distillation column. Since very little C is generated in the first reaction and the manufacturing facility does not contain any reactors and columns an order of magnitude smaller, these equipment items are underutilized as shown in table 5.22.

As with study case II.A, raw material costs dominate the production costs for this design, as shown in table 5.23. Tables 5.19 and 5.20 show the material processing costs for the campaign. Table 5.21 shows the charges incurred for the use of equipment during the campaign. The batch size and cycle time are limited by the first reaction task.

Raw Material Costs				
Raw Material	Cost [$ / kg]	Feed [kg]	Total Cost [$]	$ / kg product
M	1.23	3.29	4.04	0.00
R2	8.85	71557.98	633288.14	4.65
R1	4.11	53464.90	219740.75	1.61
W	0.01	1071.74	10.72	0.00
T	1.46	1472.62	2155.92	0.02
Total		127570.54	855199.57	6.28

Table 5.19: Raw material costs for the entire campaign for the process requiring three reaction tasks.

Utility Costs			
Cut Material	Amount [kg]	Reboiler Cost [$]	$ / kg product
Distillation 1			
CM	339.60	0.00	0.00
RT	18716.19	0.09	0.00
T	64316.76	0.18	0.00
Distillation 3			
M	158.48	0.00	0.00
WE	75.71	0.00	0.00
W	1056.22	0.02	0.00
Total	84662.96	0.29	0.00

Table 5.20: Utility costs for the distillations for the entire campaign for the process requiring three reaction tasks.

Comparison of the two superstructures

The design that requires that all three reaction tasks are performed results in higher manufacturing costs than the design with only one reaction task. Although the three reaction process has slightly lower raw material costs, this savings is outweighed by the additional equipment cost incurred by dedicating a reactor and column to the downstream processing for the duration of the campaign. Thus, the one reaction task design is more desirable if this high selectivity can be achieved through dynamic optimization of the operating policy of the first reaction.

The screening model superstructure used in this example did not consider employ-

Reactor Rental Costs					
Volume [gal]		Assigned Units	Rental Rate [$ / hr]	Total Cost [$]	$ per kg product
500		2	50	15363.07	0.11
750		1	70	10754.15	0.08
Distillation Column Rental Costs					
Volume [gal]	Vapor Rate [kmol/hr]	Assigned Units	Rental Rate [$ / hr]	Total Cost [$]	$ per kg product
750	15	2	90	27653.52	0.20
1000	20	1	110	16899.38	0.13
Total for reactors and columns				70670.12	0.52

Table 5.21: Equipment costs for the entire campaign for the process requiring three reaction tasks.

Utilization Measure	Processing Task				
	Rxn 1	Dist 1	Rxn 2	Rxn 3	Dist 3
Cycle Time	4.27	3.79	1.29	2.25	1.73
Volume Required	1246.65	1246.65	2.66	12.48	12.48
Volume Assigned	1250.00	1500.00	500.00	500.00	1000.00

Table 5.22: Equipment utilization for the design obtained from the process requiring three reaction tasks.

Cost Contributions			
Component	Percent	Total Cost [$]	$ / kg product
Raw Material	92.37	855199.57	6.28
Waste Disposal	0.00	0.00	0.00
Utility	0.00	0.29	0.00
Equipment	7.63	70670.12	0.52
Total		925869.98	6.80

Table 5.23: Comparison of raw material, waste disposal, utility, and equipment costs for the process containing only one reaction task.

ing intermediate storage or the possibility of changing the operation of the process at some point during the campaign (i.e., using an item of equipment for different tasks at different times), so the downstream items of equipment are underutilized. The use of intermediate storage alone will not improve the equipment utilization much since the same task already limits both the batch size and cycle time, but if we relax the restriction that equipment items are dedicated to a particular task for the entire campaign, a process employing three reaction tasks may become more attractive. For example, if sufficient intermediate storage is available, we might consider operating the first reaction and distillation tasks as suggested by the solution of the screening model and storing the $C - M$ azeotrope until all the batches of the first two tasks are completed. At this point, the same equipment could be employed for the second and third reactions and final distillation. Although the use of intermediate storage is considered by the screening models formulated in chapter 3, extensions to the screening model are required to consider process that do not operate in campaign mode (i.e., those in which equipment items are not dedicated to a particular task for the duration of the campaign).

5.3.3 Case III: Disposing of Recycle Streams

This example considers the cost of disposing of recycled material at the completion of the campaign. We employ the reaction targets described above and consider the process containing only one reaction task. The trade off between the size of the batches and the campaign length is considered.

We assume that the amount of material recycled per batch must be disposed of at the conclusion of the campaign, unless this material is one of the raw materials used by the process. The cost of disposing of this material is added to the objective function, and the cost to manufacture 300,000 pounds of monomer is minimized.

The solution of the screening model results in a process that differs from the one obtained when the disposal of the recycle streams was not considered. This design employs 60 batches, instead of 45, to manufacture the product at a cost of \$6.63/kg. A smaller reactor is employed and the cycle time of the process is reduced, but the campaign length is increased from 191 to 214 hours. Raw material costs are identical between this design, shown in figure 5-4, and the design shown in figure 5-2. Tables 5.24–5.26 show the raw material, waste disposal, utility, and equipment rental costs. All of the waste generated results from the disposal of the recycle streams. We assume that the recycled toluene, one of the raw materials, can be reused in another process, so no cost is assessed for this recycle. Although the distillation column is larger than necessary as shown in table 5.28, using the 1000 gallon column reduces the cycle time because it has the largest vapor rate of the available columns. Table 5.29 shows breakdown of the processing costs, demonstrating that the raw material costs still dominate.

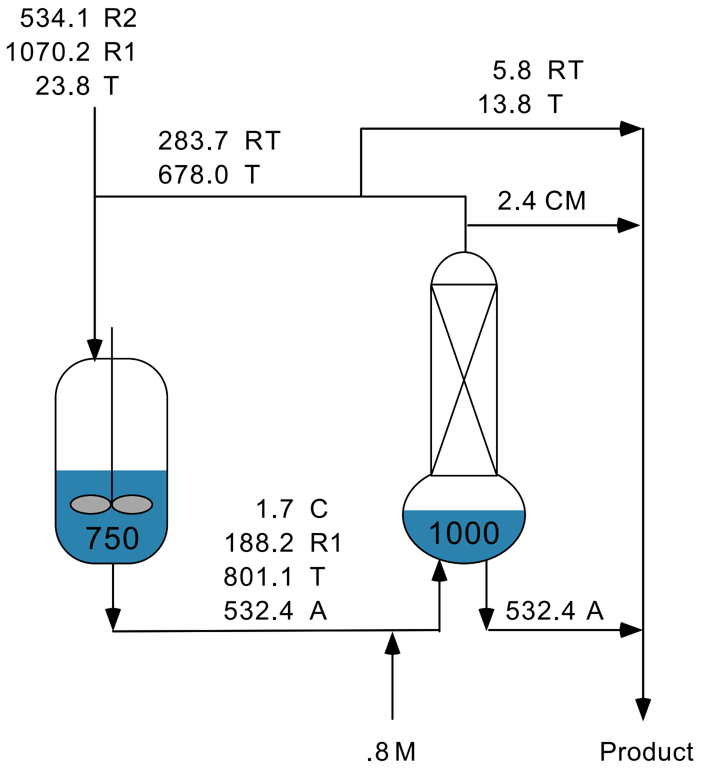

Figure 5-4: Process schematic of the solution derived from the superstructure containing only one reaction task in which the disposal of recycle streams at the end of the campaign is considered. Stream labels indicate the fixed point flows for the campaign given in kmols.

5.4 Conclusions

Computationally tractable models can be derived that provide bounds on the cost of manufacture for processes commonly encountered by synthetic pharmaceutical and

Raw Material Costs				
Raw Material	Cost [$ / kg]	Feed [kg]	Total Cost [$]	$ / kg product
M	1.23	25.94	31.91	0.00
R2	8.85	71738.25	634883.47	4.67
R1	4.11	53598.21	220288.65	1.62
W	0.01	0.00	0.00	0.00
T	1.46	2186.96	3201.71	0.02
Total		127549.36	858405.75	6.31

Table 5.24: Raw material costs for the process considering the disposal of recycled material at the completion of the campaign.

Waste Disposal Costs				
Waste Material	Cost [$ / kg]	Amount [kg]	Total Cost [$]	$ / kg product
RT	18.00	306.46	5516.28	0.04
Total		306.46	5516.28	0.04

Table 5.25: Waste disposal costs for the process considering the disposal of recycled material at the completion of the campaign.

Utility Costs			
Cut Material	Amount [kg]	Reboiler Cost [$]	$ / kg product
Distillation 1			
CM	346.09	0.00	0.00
RT	18762.85	0.09	0.00
T	64479.02	0.18	0.00
Total	83587.97	0.27	0.00

Table 5.26: Utility costs for the distillation task in the process considering the disposal of recycled material at the completion of the campaign.

specialty chemical manufacturers. These models embody many of the processing limitations governing the process design, yet they are able to consider continuous and discrete aspects of the design simultaneously. They also enable some of the process synthesis decisions to be systematically considered during the design procedure. The screening models do not consider the process dynamics, so these models must be used

Reactor Rental Costs					
Volume [gal]		Assigned Units	Rental Rate [$ / hr]	Total Cost [$]	$ per kg product
750		1	70	14978.44	0.11
Distillation Column Rental Costs					
Volume [gal]	Vapor Rate [kmol/hr]	Assigned Units	Rental Rate [$ / hr]	Total Cost [$]	$ per kg product
1000	20	1	110	23537.55	0.17
Total for reactors and columns				38515.99	0.28

Table 5.27: Equipment costs for the process considering the disposal of recycled material at the completion of the campaign.

Utilization Measure	Processing Task	
	Reaction 1	Distillation 1
Cycle Time	3.57	3.57
Volume Required	749.73	749.87
Volume Assigned	750.00	1000.00

Table 5.28: Equipment utilization for the process considering the disposal of recycled material at the completion of the campaign.

Cost Contributions			
Component	Percent	Total Cost [$]	$ / kg product
Raw Material	95.12	858405.75	6.31
Waste Disposal	0.61	5516.28	0.04
Utility	0.00	0.27	0.00
Equipment	4.27	38515.99	0.28
Total		902438.28	6.63

Table 5.29: Comparison of raw material, waste disposal, utility, and equipment costs for the process considering the disposal of recycled material at the completion of the campaign.

in conjunction with detailed dynamic simulation or pilot plant experiments. However, the solution of the screening models facilitates the cyclic steady state simulation of a dynamic process containing recycles and the formulation of a multi-stage dynamic optimization of the process by providing both initial estimates of the flowrates in the process and alternative process structures.

The solution of the process development example demonstrates that integrated processes employing recycles can significantly reduce the waste generated during the manufacture of these products. The process operates at cyclic steady state, so the recycled material does not accumulate. However, at the conclusion of the campaign, this material must either be stored indefinitely, or sent to a recovery facility. As demonstrated by the process development example the amounts that are recycled can be on the same order as the total waste generated during the campaign. The end effects of the campaign are important from the standpoint of pollution prevention and may possibly impact the design from a cost standpoint as well. Section 5.3.3 shows that if the number of batches is not very large, the cost of waste disposal at the conclusion of the campaign can affect the way in which the process is operated, trading off operating and waste costs.

As in chapter 4, the screening models demonstrate the ability to perform some aspects of the process synthesis. In fact, the results of the case study II demonstrate that the process employing only one reaction task is potentially more efficient than one that contains the downstream processing to convert C to D.[5] However, detailed dynamic models are required to perform an accurate comparison of the costs, but the solutions of the screening model provides good initial guesses for a material states involved in the dynamic optimization of the process performance.

This chapter also highlights the need to extend the screening formulations to handle both reactive distillation processes and heterogeneous mixtures. These examples assume that reaction does not occur in the distillation columns, although some reaction must occur. This was not a limitation in chapter 4 since the reactions employed a heterogeneous catalyst which was filtered before entering the distillation columns.

[5] A complete comparison requires the detailed design, but the one reaction process will be more efficient provided that a sufficiently high conversion of A versus C can be achieved using detailed dynamic optimization of the reaction I operating policy.

Chapter 6

Numerical Issues in the Simulation and Optimization of Hybrid Dynamic Systems

Section 1.6 described the need to employ hybrid discrete/continuous modeling environments for the detailed simulation and optimization of batch processes. A key to the application of modeling technology to the design of batch processes has been the evolution of equation-based simulation tools, such as SpeedUp (AspenTech, 1993), ASCEND (Westerberg *et al.*, 1994), POLYRED (Ray, 1993), or ABACUSS (Barton, 1992), into *process modeling environments* in which a common reusable process model may be used reliably for a variety of different computational tasks (Pantelides and Barton, 1993), such as both steady-state and dynamic simulation, optimization, sensitivity analysis, uncertainty analysis, etc. Such environments decouple the description of the process model from the solution procedure, yielding major advantages for the user of the system. The user is free to concentrate on the correct formulation of the model and simulation experiment rather than the details of the numerical solution procedures; thus, the user need not be an expert in numerical analysis. While this is a desirable goal, it places stringent demands and high expectations on the robustness, accuracy, and generality of the solution procedures. For example, our experience with the application of the state-of-the-art numerical algorithms employed within ABA-

CUSS to the batch distillation of wide-boiling azeotropic mixtures has demonstrated that the numerical technologies have not yet attained the level of robustness required for the routine simulation and optimization of batch processes.

The following chapters focus on improvements to the robustness and efficiency of the numerical algorithms employed within the ABACUSS process simulator. Two main areas have been investigated: 1) improving the accuracy and robustness of the integration procedure for models that become locally ill-conditioned during the course of the transient, and 2) improving the efficiency of the integration algorithm during the initial phase of the integration procedure. These improvements have been incorporated within an integration code designed for the integration of large sparse unstructured systems of differential-algebraic equations called DSL48S (Feehery *et al.*, 1997). Therefore, although the development of these techniques has been motivated by the needs of hybrid discrete/continuous simulation environments, the techniques apply to general sparse unstructured systems of DAEs.

6.1 Accuracy of Solution Procedures

Mathematical models provide a formalism with which to encapsulate our understanding of the physical world and apply this knowledge to calculations of engineering interest. The derivation of useful models comprises two tasks: a) identifying the physical phenomena relevant to the current engineering activity, and b) accurately representing this phenomena within the mathematical formalism. Identifying the relevant phenomena permits the model to capture important behavior in the physical process without obscuring the results in a sea of detail and without burdening the computation with unnecessary calculations. Accurately capturing the relevant phenomena within the mathematical model is critical to the utility of the simulation results. The derivation of good models remains a difficult task, but process modeling environments provide a framework in which to apply these models to a variety of engineering calculations. In fact, a single reusable mathematical model can be employed for engineering calculations performed over the lifetime of a process (Barton, 1992). However, the user of such an environment expects the results provided from all simulations to meet certain minimal accuracy requirements. While any user recognizes that the numerical solution is an approximation of the exact solution of the mathematical model, the solution should be a good approximation to the exact solution.

The first question to ask is how should the quality of the numerical solution be measured. In most cases, a numerical approximation that is close to the exact solution is desired; letting x^* define the exact solution and x define its numerical approximation, a close solution is one that satisfies $\|x^* - x\|_\alpha < \tau$ where τ is the tolerance. This definition also requires specification of the norm, which could be the maximum norm, the two norm, or any other norm that is desired. The norm reflects both the knowledge about the expected solution (e.g., are all the variables on the same scale?) and any requirements that the solution should satisfy (e.g., should some average property be enforced, or does every component of the solution need to satisfy a requirement in order to employ the solution for engineering purposes). The norm should also indicate whether we require relative accuracy in the solution or whether we require that some absolute tolerances are achieved. The difference between the exact and the calculated solution is referred to as the *forward error* of the solution. Usually, a small forward error would satisfy our expectations. However, in other cases we may require that we have found a solution that achieves small residuals. For instance in an interpolation problem we are likely to be more interested in whether the solution provides a good approximation of the data (either in an absolute or relative sense) rather than how well it approximates the exact solution of the problem. In many cases, bounds relating these quantities are easily derived (Higham, 1996).

Differences between the numerical solution of our mathematical model and the performance of the process being modeled come from several sources:

1. Approximations made during the abstraction of the physical process into a mathematical model.

2. Errors in the problem data. These errors may be attributed to imprecise mea-

surements of physical quantities (e.g., VLE or property measurements), the error in a previous calculation (e.g., parameter estimation), or they may simply be the result of representing an exact quantity in finite precision.

3. Truncation error arising from terminating an exact approximation (such as a Taylor series) after a finite number of terms. In many cases, the truncation error is a function of the discretization (i.e., the step size and order of a numerical integration).

4. Rounding errors arising from the fact that the computations are carried out on machines of finite precision.

The users of process modeling environments typically expect that the applicability of their simulation results depends on the errors attributed to the abstraction of the physical process, and the errors in the measured data incorporated in the model such as the parameters employed to predict physical properties. It is the user's duty to make certain that these approximations are valid and apply the results with an understanding of the potential inaccuracies. Some process modeling environments ease the interpretation of uncertainty in the problem data by calculating the sensitivity of the results to perturbations in the problem data (Barton and Galán, 1997; Tatang, 1995). The user expects the contributions of the other error components to be controlled by the numerical solution procedure to achieve the requested accuracy. The user indicates the desired solution accuracy by specifying the tolerance for the computations. This tolerance is then typically used to control the truncation error, balancing the speed of computation with the need for accuracy.

While simulating the batch distillation of wide-boiling azeotropic mixtures, we have uncovered situations where the implicit assumption that the effect of rounding errors is negligible certainly breaks down; figure 6-1 provides a dramatic illustration of this phenomenon. While the simulation results appear to predict the dominant processing characteristics correctly (ignoring the spikes), large contributions of the rounding errors were witnessed as spikes in the values of certain variables, without any accompanying warnings being issued by the numerical routines (except in cases where the algorithms simply failed). This highlights a major problem for the application of the results. The results clearly do not meet the desired accuracy requirements, but the numerical procedures do not provide any indication that this has occurred. The uninformed user may then go on to employ these results as if they were correct. Since detailed dynamic models of chemical processes are employed for the design of operating policies (Ochs *et al.*, 1996), control strategies (Zitney *et al.*, 1995), and the specification of equipment (Naess *et al.*, 1993), the application of incorrect results can waste money. Even worse, these results may be used to verify the safety of proposed operating procedures, or the safety of the process under abnormal operating conditions (Sedes, 1995). Although we have not encountered situations where these errors have changed the qualitative behavior of the simulation, it is not hard to imagine that the perturbations of the variables that have been witnessed could cause the improper identification of state events, changing the functional form of the model and leading to very different qualitative behavior (Park and Barton, 1996). In other cases, the breakdown in the control of the accuracy is not signaled by a large deviation in a variable value, but rather a failed simulation. While this result is also not desirable, at least the results are not likely to be interpreted as if they are correct.

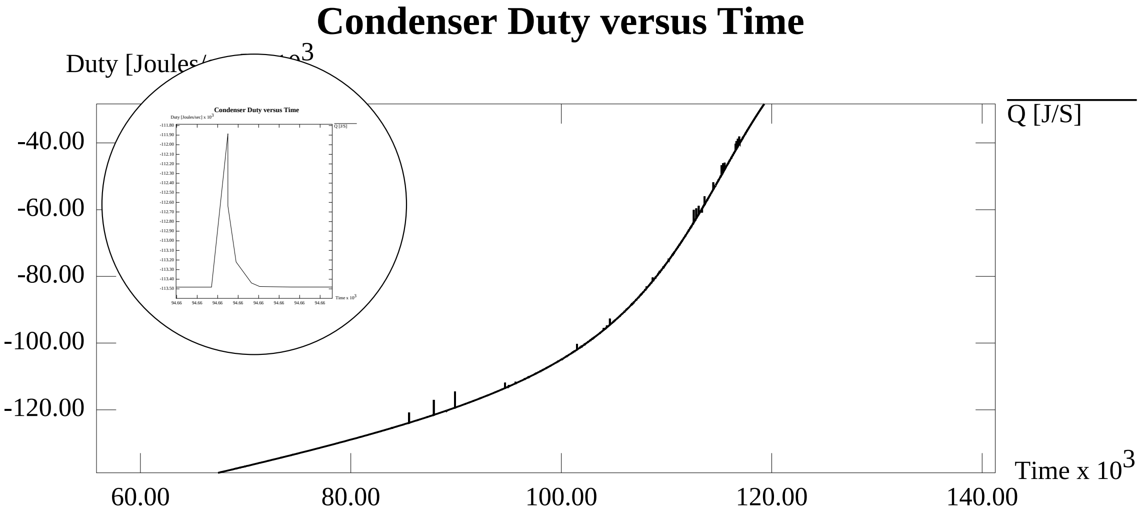

Figure 6-1: Plot of condenser duty resulting from ABACUSS simulation showing one ‘spike’ in detail.

It is unreasonable to expect that any level of accuracy can be achieved using finite precision computations.[1] However, numerical algorithms should make attempts to mitigate the effects of rounding errors (many effective algorithms are backward stable, guaranteeing a solution with small backward error), and warn users when the desired accuracy has not been maintained due to the effects of rounding error and the conditioning of the problem. As we shall prove in chapter 7, the problems we have observed are the result of ill-conditioning. We have found that automatic scaling of the equations and variables during the integration procedure improves the performance of the numerical algorithms and permits evaluation of the accuracy of the solution. Not only does this allow the computation to maintain the desired accuracy, but also improves the robustness and efficiency of the method. Before addressing the results contained in chapter 7, some background on conditioning and linear error analysis may prove useful.

6.1.1 Backward Error and Conditioning

Finite precision arithmetic imposes barriers on the accuracy that can be achieved due to the effects of the rounding errors. Even if the computations could be carried out exactly, rounding errors are encountered merely by representing the problem data in finite precision. Wilkinson (1963) recognized that the solution obtained by a numerical calculation in finite precision arithmetic is equivalent to the exact solution of a similar problem with perturbed data; the size of these perturbations is termed the *backward error*. The backward error interprets the errors committed during the calculation as perturbations of the problem data. Since errors in the problem data are encountered just from storing the problem, if the backward error is of that order we can hope to do no better during the calculation. The second motivation for bounding the backward error is that the relationship between the backward and forward errors of the problem can be determined from perturbation theory. Perturbation theory is understood for many problems (Stewart and Sun, 1990); an advantage of perturbation analysis is that it is a characteristic of the problem and not the algorithm.

[1] It is assumed that the computations are employing the machine's standard arithmetic operations and are not simulating arithmetic of arbitrarily high precision (Higham, 1996).

Backward error analysis possesses advantages over direct round-off analysis, where each algebraic computation is regarded as an operation which approximates the true algebraic process. By using the backward or inverse round-off analysis, the analysis of the solution procedure can be undertaken assuming the standard algebraic axioms. In contrast, in direct round-off analysis the multiplication and addition operations do not follow either the associative or distributive laws. Thus, an entirely different system of analysis must be devised.

The relationship between the forward and backward errors is given by the conditioning of the problem. The conditioning of a problem measures the sensitivity of the solution of the problem to perturbations in the problem data, so it is a function of the problem and not the solution algorithm. For scalar functions, the relative *condition number* measures the relative change in the output caused by a relative change in the input. For vector functions the changes are measured using a suitable norm, and the condition number measures the maximum relative change in the output caused by a relative change in the input. The maximum change in the output is achieved by some, but not all, input perturbations. When the forward error, backward error, and the condition number are defined in a consistent fashion, the following rule of thumb (Higham, 1996) demonstrates that an ill-conditioned problem can lead to a large forward error even if small backward is achieved:

$$\text{forward error} \lesssim \text{condition number} \times \text{backward error} \tag{6.1}$$

The conditioning of the linear systems solved during the corrector iteration of the BDF code indicate that large error in the values of some of the variables can be obtained even when the residuals are evaluated accurately, and the Gaussian elimination produces small backward error. Rounding error analysis and conditioning of linear systems is reviewed in section 6.3.3.

6.2 Efficiency of Integration Codes

The routine simulation and optimization of large DAE models containing discontinuities will only be realized once the solution algorithms and computer hardware enable these calculations to be performed in reasonable time on desktop workstations. When using BDF integration codes (see section 6.3.1), the computation time of the solution algorithm is dominated by the time spent factoring the corrector iteration matrix. Thus, the number of times the matrix is factored and the efficiency of the linear algebra used to factor the matrix dictate the efficiency of the BDF code. Numerical analysts have devoted years of effort developing efficient codes to factor the large sparse unstructured matrices that are obtained during the dynamic simulation of chemical processes (Duff and Reid, 1993; Zitney, 1992; Zitney and Stadtherr, 1993; Zitney *et al.*, 1996), so these algorithms will not be examined here. The heuristics employed within the implementation of the BDF method contained in a particular code typically seek to minimize the number of times the corrector iteration matrix is factored. Since the need to factor the matrix depends on the changes in the variable values and the change in the step size, it is important that the step size is on scale for the problem.

This thesis proposes two techniques to keep the step size on scale for the problem. First, the automatic scaling technique described in chapter 7 mitigates the effects of ill-conditioned models in order to avoid situations in which the step size is cut unnecessarily due to inaccurate solutions of the corrector. In addition, chapter 8 develops a method to determine an initial step size that is on scale for the problem which is required at the start of the simulation or following any discontinuity. Although both techniques benefit all dynamic models, the second technique is most applicable to simulation and optimization of hybrid dynamic systems because these calculations require the integration code to be started many times during a single simulation or optimization experiment.

6.3 Mathematical Background

Since the focus of this thesis is the application of mathematical modeling technology to the design of batch processes, the reader is likely to be more interested in the benefits provided by improvements to the numerical algorithms than the details of the numerical analysis required to develop the new solution procedures. However, some details of the numerical analysis are required to understand both the motivation and the application of the techniques developed in the following chapters. This section describes the components of the integration algorithm on which our numerical advances have focused, and provides background that is required to understand the following chapters for the reader who has not devoted a career to numerical analysis.

6.3.1 BDF Integration Codes

Backward differentiation formula (BDF) methods are a class of linear multistep methods suitable for the solution of stiff ODE systems and index-1 DAEs (Gear, 1971). In particular, BDF methods can solve DAEs expressed in fully implicit form (6.2) directly.

$$f(\dot{z}, z, t) = 0 \tag{6.2}$$

The kth order BDF method approximates the time derivative of the solution $\dot{z}(t)$ using the derivative of a kth order polynomial that approximates the solution $z(t)$ over the last $k+1$ points (including the current point). The simplest BDF method is equivalent to the implicit Euler method in which $\dot{z}$ is replaced by the first order backward difference approximation. This reduces the system of equations that must be satisfied at every time step to the following:

$$f(\frac{z_n - z_{n-1}}{h_{n-1}}, z_n, t_n) = 0 \tag{6.3}$$

where $h_{n-1} = t_n - t_{n-1}$ denotes the length of the integration step and z_n denotes the numerical approximation to the solution at t_n. For higher order BDF methods, the equations solved at each time step can be written as follows (Brenan *et al.*, 1996):

$$f(\alpha z_n + \beta, z_n, t_n) = 0 \tag{6.4}$$

where α is a constant that depends on the order of the approximation and the step size, and β is a constant that contains the contributions of the solution from previous steps to the BDF approximation of $\dot{z}(t_n)$. Although many other methods have been applied to the solution of index-1 DAEs, the greatest success has been achieved from codes based on BDF methods, probably due to their large regions of absolute stability and high accuracy (Brenan *et al.*, 1996). Several texts describe the theoretical properties of these methods in detail (Lambert, 1991; Hairer and Wanner, 1993; Brenan *et al.*, 1996).

The BDF codes examined within this thesis are implemented using a predictor-corrector scheme that automatically adjusts both the step size and the order of the approximation. The BDF method requires the solution of the system of nonlinear equations given by (6.4) at each time step, which is solved using a modified version of Newton's method. BDF predictor-corrector methods employ an explicit predictor based on extrapolation of the BDF polynomial approximation of the solution to provide an initial value for the iterative procedure used to determine the solution of the nonlinear equations z_n at t_n. The equations are converged in what is referred to as the *corrector iteration*. For convenience, we define z_n^P and z_n^C as the predicted and corrected solutions; note that z_n^C is the final Newton iterate, and not the exact solution of the model equations at t_n. After z_n^C has been determined, the quality of the approximation of the derivatives over the step is evaluated. The step is accepted if the approximation, measured an approximation of the local truncation error, is good. If the approximation is poor, then the step is rejected, and the integrator attempts another step of smaller size, noting that the approximation should be exact as the step size approaches zero.

A flowchart of the BDF integration algorithm is given in figure 6-2. We will examine the calculations performed at each step in this algorithm in more detail below.

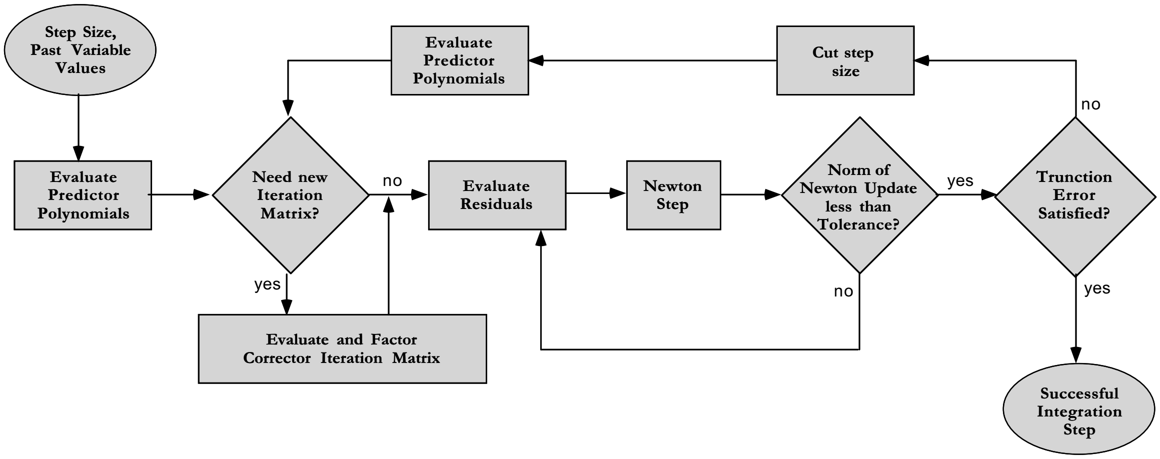
Step Size, Past Variable Values
Evaluate Predictor Polynomials
Need new Iteration Matrix?
no
yes
Evaluate and Factor Corrector Iteration Matrix
Evaluate Residuals
Newton Step
Norm of Newton Update less than Tolerance?
no
yes
Trunction Error Satisfied?
no
yes
Cut step size
Evaluate Predictor Polynomials
Successful Integration Step

Figure 6-2: Flowchart for the predictor corrector implementation of the BDF method.

Since factorization of the corrector iteration matrix is expensive, these algorithms employ the factored matrix from a previous integration step until the convergence rate of the corrector deteriorates or the step size changes substantially; either situation indicates that the factored matrix is providing a poor approximation to the current iteration matrix.

Our analysis of the BDF method focuses on the solution of the nonlinear equations performed by the corrector iteration. We will also examine the truncation error criteria to see how these criteria can be satisfied when the corrector has been converged numerically, yet z_n^C may still be far from the exact solution of the BDF equations at t_n. However, we will not discuss the theory justifying the use of an approximation to the local truncation error to control the error in the time evolution of the system; for this, the reader is referred to other texts (Lambert, 1991; Hairer and Wanner, 1993; Brenan *et al.*, 1996).

Corrector Iteration

The corrector step in the BDF integration method solves the model equations for the variable values, employing the BDF approximation to $\dot{z}$ at the integration points. At time t_n, the system of equations given by (6.4) is solved using a modified version of Newton's method in which a deferred Jacobian is employed. The corrector iteration updates the value of z_n at each step of the iteration (i.e., $z_n^{(k+1)} = z_n^{(k)} + \Delta z_n^{(k)}$) using the solution of the following linear system:

$$\left[\frac{\partial f}{\partial z} + \frac{\partial f}{\partial \dot{z}}\frac{\partial \dot{z}}{\partial z}\right] \Delta z_n \approx \left[\frac{\partial f}{\partial z} + \alpha \frac{\partial f}{\partial \dot{z}}\right] \Delta z_n = -f(\alpha z_n + \beta, z_n, t_n) \tag{6.5}$$

The corrector iteration is continued until $\|\Delta z_n\|_{\mathrm{BDF}} \leq$ Tolerance. This tolerance is defined to be small enough so that the error incurred from terminating the Newton iteration[2] will not be so large as to adversely affect the truncation error check. For example, the heuristics within DASSL require that the Newton iteration is converged to within a tolerance that is one third the size of the permissible truncation error (Brenan *et al.*, 1996).

Truncation Error Tolerance

The local truncation error is used to measure the accuracy of the backward difference approximation to the derivatives. DASSL also enforces a bound on the interpolation error – the error in the solution interpolated between the integration points. DASSL estimates the truncation error using the principle term in the infinite series expansion of the local truncation error (Brenan *et al.*, 1996). The interpolation error is estimated in a similar fashion. Both DASSL and DASOLV (Jarvis and Pantelides, 1991) require that the following condition is satisfied before an integration step is accepted (Brenan *et al.*, 1996):

$$\text{error} = M \cdot \left\|z^C - z^P\right\|_{\mathrm{BDF}} \leq 1.0 \tag{6.6}$$

where z^C is the corrected solution, z^P is the predicted solution and M is a constant that depends on the order of approximation and the current step size. The user requested integration tolerances are buried in the definition of the norm employed in (6.6). Let $\|\cdot\|_{\mathrm{BDF}}$ represent default norm used by the BDF integration routines to measure the truncation error and size of the corrector updates. It is defined in (6.7),

[2] This error is also commonly referred to as truncation error, the error from truncating the infinite series of Newton iterates after a finite number of iterations, but we will simply refer to it as the termination error to avoid confusion with the local truncation error.

where z_i^p is the value of the variable z_i from the previous integration step, τ_{ri} is the relative error tolerance and τ_{ai} is the absolute error tolerance for variable i (Brenan *et al.*, 1996).

$$\|\mathbf{z}\|_{\mathrm{BDF}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left|\frac{z_i}{\tau_{ri}\,|z_i^p| + \tau_{ai}}\right|^2} \tag{6.7}$$

Section 7.2.2 discusses this truncation error criterion (6.6) in more detail and explains how it permits the generation of 'spikes' in the solution trajectory.

6.3.2 Dynamic Optimization

The performance subproblem described in section 2.4 defines a dynamic optimization problem. The goal is to determine the operating policies for the tasks that minimize the operating cost for a fixed allocation of the plant's resources. A relatively general formulation for the dynamic optimization problem can be stated as follows:

$$\min_{u(t),v,t_f}\left\{\phi(z(t_f),u(t_f),v,t_f) + \int_{t_0}^{t_f} L(z(t),u(t),v,t)dt\right\} \tag{6.8}$$

Subject to:

$$\begin{aligned}
f(z(t),\dot{z}(t),u(t),v,t) &= 0 \quad \forall t \in [t_0,t_f] && (6.9)\\
g(z(t),\dot{z}(t),u(t),v,t) &\leq 0 \quad \forall t \in [t_0,t_f] && (6.10)\\
k_p(z(t_p),\dot{z}(t_p),u(t_p),v,t_p) &\leq 0 \quad \forall p \in \{0,n_p\} && (6.11)
\end{aligned}$$

where

$$z \in Z \subseteq \mathbb{R}^{n_z} \quad u \in U \subseteq \mathbb{R}^{n_u} \quad v \in V \subseteq \mathbb{R}^{n_v}$$

$$f : Z \times \mathbb{R}^{n_z} \times U \times V \times \mathbb{R} \to \mathbb{R}^{n_z}$$

$$g : Z \times \mathbb{R}^{n_z} \times U \times V \times \mathbb{R} \to \mathbb{R}^{n_g}$$

$$k_p : Z \times \mathbb{R}^{n_x} \times U \times V \times \mathbb{R} \to \mathbb{R}^{n_{k_p}}$$

and $z(t)$ are the continuous variables describing the state of the dynamic system, $u(t)$ are the controls whose optimal time variations on the interval $[t_0, t_f]$ are required, v are time invariant parameters whose optimal values are also required, and t_f is a special time invariant parameter known as the final time. Equation (6.9) represents a general set of differential-algebraic equations (DAEs) describing the dynamic system. As such, they will include a lumped dynamic model of the system in question coupled with any path equality constraints the system must satisfy; the number of controls that remain as decision variables in the optimization is reduced by each path equality constraint added to the formulation; we assume that (6.9) defines a solvable DAE. However, the choice of controls $u(t)$ and the presence of path constraints may have a profound influence on the *differential index* (Brenan *et al.*, 1996) of (6.9). For practical purposes, we will further assume that, while (6.9) may have arbitrary index, the index is time invariant and both the index and the dynamic degrees of freedom can be correctly determined using structural criteria. Hence, the method of dummy derivatives may be used either for numerical solution of the initial value problems (IVPs) in (6.9) (Mattsson and Söderlind, 1993; Feehery and Barton, 1996a), or to derive an equivalent index-1 discretization of (6.9) via collocation (Feehery and Barton, 1995).

Solving Dynamic Optimization Problems

Two approaches that have been applied to the numerical solution of dynamic optimization problems are discussed here. The traditional approach (Pontryagin *et al.*, 1962) employs the classical necessary conditions for optimality derived from the calculus of variations directly. This formulation of the problem requires the solution of a two-point boundary value problem (TPBVP). Although this results in an mathematically elegant formulation, numerical solution of the resulting TPBVP is difficult, particularly when the controls appear linearly in (6.9) or inequality path constraints (6.10) are imposed on the state variables. A more practical approach is to transform the variational problem into a nonlinear program (NLP) and then solve the NLP using standard codes. This approach has been applied successfully to some fairly large problems (Mujtaba and Macchietto, 1993; Charalambides, 1996).

Two methods, control vector parameterization (Kraft, 1985) and collocation (Logsdon and Biegler, 1989), have been used to transform the DO problem into a NLP. The resulting NLPs differ in both form and size, but the conditions defining a local optima of the NLPs correspond to the classical necessary conditions for the dynamic optimization (Bryson and Ho, 1975). The first approach approximates the control variables using functions defined in terms of a finite number of parameters that are the decision variables of the NLP (Sargent and Sullivan, 1977; Morison and Sargent, 1986; Vassiliadis, 1993). The objective function is evaluated by solving the initial value problem, and the function gradients are calculated by augmenting the DAE system with the equations defining the parametric sensitivities and solving the resulting initial value problem. In this approach, the discretization of the control variables is defined during the problem formulation, but the discretization of the state variables of the DAE, which controls the accuracy of the solution to the dynamic model, is determined automatically during solution of the IVP. On the other hand, the collocation approach discretizes the state and control variables simultaneously. The NLP is used to solve the optimization and the simulation at the same time (Logsdon and Biegler, 1989; Vasantharajan and Biegler, 1990; Tanartkit and Biegler, 1995).

Although both approaches have advantages and disadvantages, the control vector parameterization approach appears to be more practical for the types of problems in which we are interested for several reasons. First, the method can be implemented directly within equation-based simulation environments so that the same models of the processing tasks and the same integration codes can be used for simulation and optimization (Barton *et al.*, 1996).[3] The approach also automatically controls the accuracy of the solution to the DAE model. Finally, the resulting NLP is much smaller since the only decision variables are the parameters defining the control variables. Although the problem size may impose the greatest barrier to the implementation of the collocation approach, the inability to control the accuracy of the DAE solu-

[3] We note that the dynamic optimization cannot yet handle implicitly discontinuous models although the simulation can.

tion automatically during the NLP begs the question of whether the results of the optimization are meaningful.

This thesis employs the control vector parameterization approach to dynamic optimization that has been implemented within ABACUSS. A schematic of the implementation is shown in figure 6-3. The implementation uses Lagrange polynomials, defined on finite elements, to specify the control functions. The user is free to specify the number of finite elements, the order of the polynomial approximation, and whether the controls should be continuous across finite element boundaries. Note that when the dynamic model decomposes into subsystems in which no dynamic interactions between the subsystems exist (e.g., (6.13–6.14), the initial value problems for each subsystem can be solved independently.

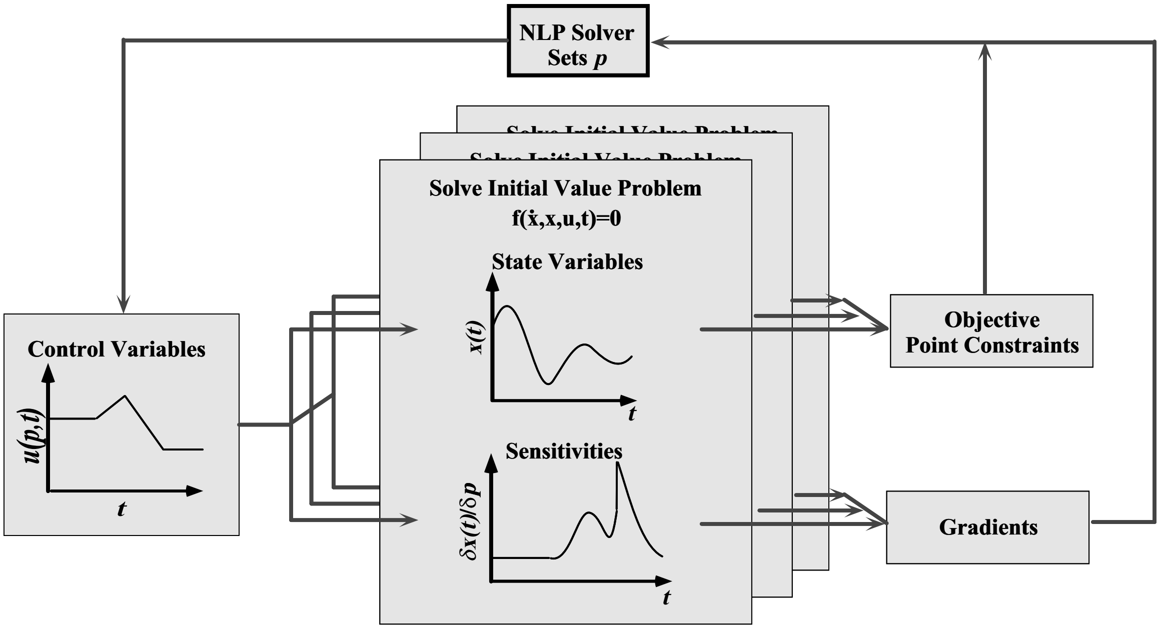

Figure 6-3: Implementation of the dynamic optimization algorithm within ABACUSS.

Dynamic Optimization of Batch Processes

For the optimization of batch processes using control vector parameterization, a slightly different form of the dynamic optimization problem is sometimes preferred than the one given by (6.8–6.11). If the dynamic interactions between processing tasks can be safely ignored, and if the process is operating at cyclic steady state, then the interactions between different processing tasks can be decoupled through the state of the material that is transferred between the tasks. These states do not change from batch to batch, so they can be represented using a subset of the time-invariant parameters v appearing in the original formulation. This allows us to partition both the equations and the variables in the formulation given by (6.8–6.11) according to the tasks with which both are associated; these tasks are identified by the subscript k. We introduce an additional set of time invariant parameters $t_k^f \leq t^f$ to denote the final time of each of the k tasks. We choose not to partition the time invariant parameters, noting that some parameters are associated with more than one task in order to obtain the following alternative dynamic optimization formulation:

$$\min_{u(t),v,t_f} \left\{ \left(\sum_k \phi(x_k(t_k^f), u_k(t_k^f), v, t_k^f) + \int_{t_0}^{t_k^f} L(x_k(t), u_k(t), v, t)dt \right) \right\} \tag{6.12}$$

Subject to:

$$\begin{aligned} f_k(x_k(t), \dot{x}_k(t), u_k(t), v, t) &= 0 \quad \forall k, t \in [t_0, t_k^f] && (6.13) \\ g_k(x_k(t), \dot{x}_k(t), u_k(t), v, t) &\leq 0 \quad \forall k, t \in [t_0, t_k^f] && (6.14) \\ k_p(x(t_p), \dot{x}(t_p), u(t_p), v, t_p) &\leq 0 \quad \forall p \in \{0, n_p\} && (6.15) \end{aligned}$$

Note that the point constraints do not partition the variables into the k subsets, since these constraints are used to relate the parameters in multiple tasks (e.g., a parameter may represent the effluent rate from one task, which must be equal to another parameter representing the charge rate to another task).

A couple of reasons exist for formulating the problem in this fashion. First, the integration of each of the k DAE systems can be performed separately, facilitating the application of parallel computation. It also reduces the computational effort required to integrate the DAE and the associated sensitivity equations on single processor machines. Although significant savings may be obtained because each system is smaller, any decent linear algebra routines would also recognize this structure of the original system, and factor the overall system as a sequence of blocks (Harwell, 1993). However, significant additional benefits are achieved because the dynamic interactions between tasks are not important, so each task k can employ a different sequence of step sizes to control the truncation error of only those variables appearing in task k. For example, consider a batch reactor and a batch distillation column. For the purpose of illustration, assume that rapid transients exist in the reactor during the initial phase of the reaction, requiring small integration steps to maintain accuracy. If the column is in the midst of a product cut at the same time, then the compositions and temperatures within the column are changing slowly. When the two tasks are integrated separately, the column is able to take large integration steps during this period; however, when they are integrated together, the step size is restricted to maintain accuracy of the reactor's variables. The opposite situation arises if the column contains rapid transients because it is near the end of a product cut, but the reaction is nearly completed and possesses transients that are slow. Integrated separately, the reactor can take large steps, but integrated together, small steps must be taken. Hence, by integrating the problems separately, the number of integration steps that must be taken to simulate each problem is reduced.

The second reason for expressing the optimization in this form is because it introduced the additional time invariant parameters t_k^f, permitting each task to operate for a different length of time. If the dynamic optimization considers a single processing train (i.e., no intermediate storage between tasks), then the difference $t^f - t_k^f$ defines the idle time of task k. This formulation attempts to make up for the fact that current implementation of control vector parameterization cannot handle discrete changes to the models, which makes it difficult to model the idling of many of the processing tasks. For example, the equations modeling the batch distillation may not apply if the column is sitting idle. When the column is idle, the vapor flow in the column goes to zero. This changes the equations governing the hydrodynamics in the tray section (actually the hydrodynamics change dramatically well before the vapor flow gets to zero (Kister, 1990)). Thus, the optimization must either handle models that can represent both hydrodynamic regimes, or the optimization must deal with idle tasks in a different fashion. This technique for dealing with the idle tasks can be viewed as a work around. Clearly, the dynamic optimization would be far more applicable if general discrete/continuous models of the processing tasks could be employed. For instance, discontinuous models are often used to define the physical properties of the components in the system. For example, the Antoine vapor pressure equation is only valid over a limited temperature range, and a different correlation is used to extrapolate outside that temperature range (Reid *et al.*, 1987). While the ability to handle discontinuous models is not currently implemented, recent theoretical developments permit the transfer of the parametric sensitivities across implicit discontinuities (Barton, 1996), so a practical implementation to optimize DAE models with implicit discontinuities will be achieved soon.

Charalambides (1996) choose to formulate and solve the performance subproblems encountered during batch process development according to the formulation given by (6.12–6.15). He notes that the number of optimization parameters can be reduced by exploiting the fact that for sequences of tasks without recycles feeds to the downstream tasks are entirely determined by feed and operating conditions of the upstream tasks. Thus, the parameters defining the state of the feeds to the downstream tasks can be eliminated from the optimization, since these are determined by the performance of the upstream task. However, he has found that exploiting this 'state task coupling' and reducing the size of the NLP is not warranted. At each iteration of the NLP, the DAE model along with the associated sensitivity equations must be integrated. Exploiting the state task coupling does not reduce the number of sensitivity equations; in fact, the sensitivity equations for the downstream models are simply defined with respect to the upstream parameter when the parameter associated with the downstream model is eliminated. Therefore, exploiting the state task coupling does not reduce the effort required to solve the IVPs. On the other hand, exploiting state task coupling will reduce the size of the NLP. However, Charalambides notes that the effort required for the solution of each IVP is far greater than that required for solving the quadratic programming subproblem used to determine the updates of the optimization parameters. He argues that only small savings could be achieved by eliminating the intermediate parameters. In addition, his experience solving these problems has demonstrated that the NLP performs better when state task coupling is not exploited. He asserts that this is due to better conditioning of the NLP, since small changes to parameters associated with upstream tasks may have little effect on the performance of a task several stages downstream (Charalambides, 1996).

Dynamic optimization using control vector parameterization requires the solution of multiple initial value problems. For the formulation (6.12–6.15), the controls of every subproblem k are defined on a domain containing n_{e_k} finite elements. An initial value problem is solved on each of these elements, where the initial conditions for the IVP of subproblem k on element e_k are defined in terms of the values of the controls and time invariant parameters associated with task k and the conditions existing at the end of the element $e-1$. Therefore, at each iteration of the NLP, N^{IVP} IVPs must be solved, where $N^{\text{IVP}} = \sum_k n_{e_k}$. Since the solution of a single dynamic optimization requires the solution of many IVPs, the solution efficiency of the IVP is important. Chapter 8 improves the efficiency of the initial phase of the integration for each of the IVPs encountered. Moreover, in order for the dynamic optimization algorithm to succeed, the solution of each initial value problem must be carried out without user intervention. Therefore, a robust IVP code is needed. This research improves the robustness of the numerical integration method used for the solution of the IVP in chapter 7.

6.3.3 Rounding Error Analysis

Determining the effect that rounding errors have on the performance of the corrector iteration employed within the BDF integration code requires a basic understanding of the methods for analyzing the effect of rounding error, rounding error analysis for linear systems, and the properties of Newton's method. This section reviews some of the basic concepts that are exploited in the following chapters.

The calculation of each Newton update requires the solution of the system of linear equations. In order to examine the performance of Newton's method in the presence of rounding error, we first review the error analysis typically applied when solving a linear system of equations on a computer using a floating point number system.

Linear Error Analysis

To ease the notation, consider the linear system in (6.16) which is equivalent to (6.37) for a particular iteration.

$$Ax = b \tag{6.16}$$

Consider that the problem data, A and b, are subject to uncertainty (either from their calculation or simply from rounding the elements of A and b to store them in the computer); we need to know what effect this error has on the calculated solution x. Assume that A is known exactly and the vector b contains uncertainty. The solution obtained is the solution to the similar problem

$$A(x + \delta x) = b + \delta b \tag{6.17}$$

Since the error obeys $A\delta x = \delta b$, we can obtain a bound for the $||\delta x||$ for any nonsingular matrix A.

$$\begin{aligned} \delta x &= A^{-1}\delta b && (6.18) \\ ||\delta x|| &\leq \left\|A^{-1}\right\| ||\delta b|| && (6.19) \end{aligned}$$

In similar fashion, (6.16) imposes a bound on $||b||$ which can be combined with (6.19) to bound the relative error in x in terms of the relative error in b.

$$\begin{aligned} ||b|| &\leq ||A|| \, ||x|| && (6.20) \\ \frac{||\delta x||}{||x||} &\leq ||A|| \left\|A^{-1}\right\| \frac{||\delta b||}{||b||} && (6.21) \end{aligned}$$

For any nonsingular matrix A, the quantity $||A|| \, ||A^{-1}||$ is defined as the *condition number* of A for any consistent norm. Thus, the value of the condition number depends upon the norm on which it is defined. When the underlying norm is to be stressed, subscripts are used. We define

$$\kappa_\alpha(A) = \|A\|_\alpha \left\|A^{-1}\right\|_\alpha \tag{6.22}$$

as the condition number of A dependent upon the α-norm. For the Euclidean norm, the condition number is a measure of the maximum distortion that the linear transformation A makes on the unit sphere. Equality holds for the inequality in (6.21) if the directions b and δb are chosen appropriately, so no sharper bound is possible. In fact, choosing b in the direction of the eigenvector of A^TA corresponding to the largest singular value of A and choosing δb in the direction of the eigenvector of A^TA corresponding to the smallest singular value of A (the largest singular value of A^{-1}) leads to equality in (6.21).

The error analysis performed above makes no reference to the rounding errors that are invariably encountered at each algebraic operation during the solution of the linear system, the backward error of the solution algorithm. The preceding perturbation analysis assumed uncertainty in the initial problem data, but exact arithmetic was used to analyze the effect of this uncertainty on the solution of the problem. Next, we review the techniques employed to assess the backward error associated with the solution of a system of linear equations by Gaussian elimination.

Wilkinson (1963) has shown that the rounding error encountered during the solution of the system by Gaussian elimination is equivalent to attributing the rounding error to uncertainty in the original problem data. For instance, Forsythe and Moler (1967) demonstrated that the rounding error from the matrix factorization and back substitution (for a dense system) can be associated with an uncertainty in the original matrix A, even though error is encountered at each step of the solution procedure (e.g., storing A in finite precision with error E, then solving $A + E = LU$, $Lz = b$, and $Ux = z$). The rounding error is attributed to uncertainty in the matrix data in each step, and the sum of these uncertainties is lumped together as the uncertainty in A, denoted by δA.

$$\begin{aligned}
LU &= A + E && (6.23)\\
(L + \delta L)(U + \delta U)x &= b && (6.24)\\
(A + E + (\delta L)U + L(\delta U) + \delta L \delta U)x &= b && (6.25)\\
(A + \delta A)x &= b && (6.26)\\
\|E\|_\infty + \|\delta L\|_\infty \|U\|_\infty + \|L\|_\infty \|\delta U\|_\infty + \|\delta L\|_\infty \|\delta U\|_\infty &\equiv \|\delta A\|_\infty && (6.27)
\end{aligned}$$

Forsythe and Moler (1967) provide a bound for the quantity $\|\delta A\|_\infty$ in terms of $\|A\|_\infty$ and other quantities (e.g., the growth factor) that can be calculated during the solution process. However, they found no systems of equations which even approached this bound. Wilkinson (1963) states that $\|\delta A\|_\infty$ is rarely larger than $nu \|A\|_\infty$ where u[4] is the machine unit rounding error and n is the dimension of A, and Golub and Van Loan (1989) use this approximation of $\|\delta A\|_\infty$ in their analysis of the error in the solution of a linear system.

The theoretical bounds for the backward error encountered during Gaussian elimination with either partial or full pivoting are typically stated in terms of the growth factor. When the solution of $Ax = b$ is computed using Gaussian elimination in finite precision arithmetic, the computed solution $\hat{x}$ obeys the equation $(A + \delta A)\hat{x} = b$, where the backward error is bounded in terms of the growth factor $g(A)$ (Golub and Van Loan, 1989):

$$\|\delta A\|_\infty \leq 8n^3 g(A) \|A\|_\infty \qquad (6.28)$$

The n^3 is hardly ever seen and can be replaced by n in practice (Higham and Higham, 1989), but the theoretical bound for $g(A)$ is 2^{n-1} when partial pivoting is employed. Although bounds on the growth factor when full-pivoting is employed are tighter, matrices that approach the theoretical bounds have not been discovered, in spite of the fact that classes of real matrices exist for which a growth factor of at least $n/2$ is

[4] For floating point arithmetic using base β with t digits stored in the mantissa, $u = \beta^{-t}$.

assured (Higham and Higham, 1989). It was conjectured that the growth factor for Gaussian elimination with full pivoting was bounded by n, but a counter example was recently found (Gould, 1991; Edelman, 1992). The conclusion to be drawn from this analysis is that when a tight bound on the backward error resulting from Gaussian elimination is required, it should be calculated for the particular matrix on hand, unless the matrix has very specific structure for which tight theoretical bounds are possible.

A posteriori analysis of the backward error resulting from Gaussian elimination can be performed. Letting $\hat{L}$ and $\hat{U}$ denote the computed upper and lower triangular matrices corresponding to A, we see that the backward error δA is defined by $A + \delta A = \hat{L}\hat{U}$. While the exact calculation of $\left\| A - \hat{L}\hat{U} \right\|$ is expensive, fairly tight bounds for the backward error can be computed quite cheaply to verify the stability of the matrix factorization (Higham, 1996). In fact, for sparse matrices it has been argued that the direct computation of the backward error is inexpensive and can be performed during the elimination (Reid, 1987), so these quantities can be made available for a posteriori analysis of computed solutions, especially if the factored matrix is employed for repeated calculations, which is precisely the situation encountered with the corrector iteration matrix used by BDF integration codes. Furthermore, Arioli *et al.* (1989) have developed a method to bound the backward error for the LU factorization of sparse unstructured matrices.

It is important to note the problems we have encountered during the integration of DAEs are the result of ill-conditioning of the problem and not the result of a particular matrix that exhibits poor backward stability during Gaussian elimination[5]. Therefore, our analysis of the error in the linear systems has focused on the conditioning of the problem and not the stability of the Gaussian elimination.

[5] The solution of linear systems obtained using a backward stable algorithm (SVD) were virtually identical to those obtained from Gaussian elimination

6.3.4 Scaling of Linear Systems

Typical methods for scaling the linear system $Ax = b$ employ two nonsingular diagonal scaling matrices, D_1 and D_2, to produce a linear system in terms of transformed variables $(D_1^{-1}AD_2)y = D_1^{-1}b$. The matrix AD_2 is often referred to as a column-scaled equivalent of A, and $D_1^{-1}A$ is referred to as a row-scaled equivalent. The objective of the scaling process is to improve the quality of the computed solution of the linear system. If we select the column scaling based on other information, such as the appropriateness of the measuring the solution error in terms of that norm, then the scaling problem is reduced to the search for the optimal row scaling matrix D_1^{-1}. For the corrector iterations with which we are concerned, the way in which the error is measured is dictated by the user requested tolerances. Therefore, this thesis is concerned with the row scaling that will improve the quality of the computed solution. Row scaling techniques to improve the solution of linear systems are discussed below.

We desire a matrix D_1 that minimizes the condition of the scaled matrix $D_1^{-1}A$, where A can be regarded as the original matrix A or a matrix that has already been transformed by a column scaling to reflect the appropriate error criteria. Since the bound on the error in the computed solution is a function the backward error of the solution method (Gaussian elimination) and the condition of the matrix, we would like to reduce both. The LU factorization codes seek to reduce the backward error of the Gaussian elimination algorithm, so we focus on reducing the condition number of the system. This provides us with tighter bounds on the accuracy of the solution (the forward error), given the same backward error. To simplify the notation, we will minimize the condition number of the matrix DA, where D is a diagonal matrix. Obviously, the choice of the optimum scaling depends on the norms upon which the condition number is defined. For certain classes of norms, an optimal scaling can be found easily using row equilibration (Bauer, 1963; van der Sluis, 1969). Even though these classes do not include the two norm, we can derive bounds on the difference between the two norm condition number provided by the optimal scaling matrix obtained for one of these norms and the condition number of the optimally row scaled matrix according to the two norm; for the sparse matrices in which we are interested, we show that these bounds are tight enough to allow simple row scaling techniques to bring us very close to the best possible row scaling for an arbitrary sparse matrix.

6.3.5 Row Equilibration

van der Sluis (1969) generalizes the work of Bauer (1963), demonstrating that *row equilibration* can satisfy the optimal row scaling for a fairly wide class of norms. The following definitions are required to understand the theorems and proofs that follow. $\mathcal{M}_{mn}$ will denote the set of real or complex $m \times n$ matrices, $m \geq n$, and A will always be an member of $\mathcal{M}_{mn}$. $\mathcal{D}_m$ and $\mathcal{D}_n$ will denote the class of non-singular real or complex $m \times m$ or $n \times n$ diagonal matrices. X and Y denote real or complex metric spaces of dimension n and m with distance functions $\|\cdot\|_\omega$ and $\|\cdot\|_\alpha$ respectively. All of $\mathcal{M}_{mn}$, $\mathcal{D}_m$, $\mathcal{D}_n$, X, and Y will be real or all will be complex. This induces the quantities

$$\sup_{\alpha\omega}(A) = \max_{x \neq 0} \frac{\|Ax\|_\alpha}{\|x\|_\omega} \quad \text{and} \quad \inf_{\alpha\omega}(A) = \min_{x \neq 0} \frac{\|Ax\|_\alpha}{\|x\|_\omega}$$

for any $A \in \mathcal{M}_{mn}$.

A vector norm[6] is *absolute* if $\|x\| = \||x|\|$, and it is *monotonic* if $|x| \leq |y| \Rightarrow \|x\| \leq \|y\|$.[7] Absoluteness and monotonicity of a vector norm are equivalent (Bauer *et al.*, 1961). A vector norm is *strongly monotonic* if it is monotonic and $|x| \leq |y|$ and $|x| \neq |y| \Rightarrow \|x\| < \|y\|$. Any Hölder p-norm of index $p < \infty$ is strongly monotonic. These definitions extend to matrix functions as follows.

Definition 6.1. *A non-negative function ϕ on $\mathcal{M} \subset \mathcal{M}_{mn}$ will be called* left-, right-,

[6] Any function $d(p, q)$ defined on a metric space that has the following three properties can be considered a distance function (W. Rudin, 1976)[pg. 30]: $d(p, q)$ is positive if $p \neq q$, symmetric ($d(p, q) = d(q, p)$), and $d(p, q)$ satisfies the triangle inequality. Golub and van Loan (1989) define a vector norm according to these same properties.

[7] The $|\cdot|$ notation implies an element by element comparison of the modulus.

or two-sided monotonic *if for all $A \in \mathcal{M}$ either*

$$\mathcal{D}_m \mathcal{M} = \mathcal{M} \quad \textit{and} \quad \phi(DA) \leq \phi(A) \max |d_{ii}| \ \ \forall \ \ D \in \mathcal{M} \tag{6.29}$$

or

$$\mathcal{M}\mathcal{D}_n = \mathcal{M} \quad \textit{and} \quad \phi(AD) \leq \phi(A) \max |d_{ii}| \ \ \forall \ \ D \in \mathcal{M} \tag{6.30}$$

or both are satisfied.

van der Sluis' Th. 1.14 (1969) proves that if $\|\cdot\|_\alpha$ and $\|\cdot\|_\omega$ are Hölder norms of any index, the functions $\sup_{\alpha\omega}$ and $\inf_{\alpha\omega}$ are two-sided monotonic.

For any two matrices A and B and any two matrix functions $\phi : \mathcal{M}_{mn} \to \mathbf{R}$ and $\psi : \mathcal{M}_{mn} \to \mathbf{R}$ we define

$$\mathcal{X}(B, A) = \frac{\psi(B)}{\phi(A)} \tag{6.31}$$

if the right hand side exists. These definitions permit the statement of the row equilibration theorem (van der Sluis, 1969).

Theorem 6.1. *If $\psi(B) = \max_j \left\|(B^H)_j\right\|_\gamma$ (where $(B^H)_j$ denotes the j-th column of B^H which is the j-th row of $\bar{B}$)*[8] *and ϕ is left-monotonic on $\mathcal{D}_m A$ and $\tilde{D} \in \mathcal{D}_m$ is such that $\tilde{D}B$ is row-equilibrated in the sense of $\|\cdot\|_\gamma$ (i.e., all columns of $(\tilde{D}B)^H$ have equal γ-norm). Then*

$$\mathcal{X}(\tilde{D}B, \tilde{D}A) = \min_{D \in \mathcal{D}_m} \mathcal{X}(DB, DA) \tag{6.32}$$

Furthermore, any matrix D for which the minimum above is attained may be obtained by multiplying $\tilde{D}$ by a diagonal matrix whose diagonal elements have equal modulus if and only if ϕ is strongly left-monotonic at $\tilde{D}A$.

[8] *$\bar{B}$ denotes the matrix whose elements are the complex conjugates of the corresponding elements of B.*

The final statement of the theorem indicates that the diagonal matrix is determined from B while the uniqueness of the matrix is determined by the properties of A. The important result provided by this theorem is that row equilibration minimizes some of the commonly used matrix condition numbers obtained when A and B represent the same matrix. For convenience, define $\mathcal{X}(A) = \psi(A)/\phi(A)$. Some useful relationships are derived from the theorem for square matrices A in which $\phi(A)$ is represented by any Hölder p-norm of A^{-1}; these relationships generalize for non-square A by replacing $\|A^{-1}\|_p$ with $1/\inf(A)$ since both $\|A^{-1}\|_p$ and $1/\inf(A)$ are two-sided monotonic functions of A (van der Sluis, 1969)[Th. 1.14]. The following relationships illustrate the result of the theorem 6.1:

- $\mathcal{X}(\tilde{D}A) = \max_j \left\|((\tilde{D}A)^H)_j\right\|_2 \left\|\tilde{D}A^{-1}\right\|_p$ is minimized when the rows of $\tilde{D}A$ have equal 2-norm.
- $\mathcal{X}(\tilde{D}A) = \left\|\tilde{D}A\right\|_\infty \left\|\tilde{D}A^{-1}\right\|_p$ is minimized when the rows of $\tilde{D}A$ have equal 1-norm.
- $\mathcal{X}(\tilde{D}A) = (\max |d_i a_{ij}|) \left\|\tilde{D}A^{-1}\right\|_p$ is minimized when when the rows of $\tilde{D}A$ have equal ∞-norm.

The first relationship follows directly from theorem 6.1 when $\psi(\tilde{D}A) = \max_j \left\|((\tilde{D}A)^H)_j\right\|_2$. The second follows when $\psi(\tilde{D}A) = \max_j \left\|((\tilde{D}A)^H)_j\right\|_1 = \left\|\tilde{D}A\right\|_\infty$. The third follows when $\psi(\tilde{D}A) = \max_j \left\|((\tilde{D}A)^H)_j\right\|_\infty$.

When examining the accuracy of the corrector iteration, we are concerned with the condition number defined on the $\|\cdot\|_{\mathrm{BDF}}$, which is the two norm condition number in a transformed system of coordinates. Unfortunately, none of the row equilibrations above minimize the condition number defined on the two-norm of the matrix. However, van der Sluis (1969) has demonstrated that the two norm condition number of the optimally row scaled matrix is within a factor of $\sqrt{m}$ of two norm condition number produced by row equilibration. We prove this below. Let $\mathcal{X}(DA) = \max_j \left\|((DA)^H)_j\right\|_2 / \phi(DA)$, and let $\tilde{D}$ be a matrix that equilibrates the two norm of the rows of DA (e.g., $\left\|(\tilde{D}A)_i\right\|_2 = \left\|(\tilde{D}A)_j\right\|_2 \;\; \forall i, j$). The row equilibration theorem states the following:

$$\min_{D\in\mathcal{D}_n} \mathcal{X}(DA) = \min_{D\in\mathcal{D}_n} \frac{\max_j \left\|((DA)^H)_j\right\|_2}{\phi(DA)} = \frac{\max_j \left\|((\tilde{D}A)^H)_j\right\|_2}{\phi(\tilde{D}A)} \tag{6.33}$$

which simplifies to the following since the rows are equilibrated:

$$\min_{D\in\mathcal{D}_n} \frac{\max_j \left\|((DA)^H)_j\right\|_2}{\phi(DA)} = \frac{\left\|((\tilde{D}A)^H)_k\right\|_2}{\phi(\tilde{D}A)} \quad \forall\;\; k = 1,\ldots m \tag{6.34}$$

From the properties of matrix norms (see appendix A for proof) we know the following:

$$\min_{D\in\mathcal{D}_n} \frac{\max_j \left\|((DA)^H)_j\right\|_2}{\phi(DA)} \leq \min_{D\in\mathcal{D}_n} \frac{\|DA\|_2}{\phi(DA)} \leq \sqrt{m} \min_{D\in\mathcal{D}_n} \frac{\max_j \left\|((DA)^H)_j\right\|_2}{\phi(DA)} \tag{6.35}$$

The desired result is obtained by combining (6.34) and (6.35) to yield the following:

$$\min_{D\in\mathcal{D}_n} \frac{\|DA\|_2}{\phi(DA)} \leq \sqrt{m}\frac{\left\|((\tilde{D}A)^H)_k\right\|_2}{\phi(\tilde{D}A)} \quad \forall\;\; k = 1,\ldots m \tag{6.36}$$

In chapter 7 we extend the key result given in (6.36) to sparse unstructured matrices scaled by diagonal matrices that are integer powers of the machine base. We prove that row equilibration provides much tighter bounds for sparse matrices and that the scaling can be performed cheaply.

6.3.6 Properties of Newton's Method

Consider the mapping $f : \mathbb{R}^n \to \mathbb{R}^n$. A solution $x \in \mathbb{R}^n$ to the system of equations defined by f such that $f(x) = 0$ is desired. Let $x_0 \in \mathbb{R}^n$ denote the initial approximation to the solution of the system of equations. Newton's method attempts to improve x_0 using the iteration defined in (6.37).

$$x_{k+1} = x_k - (\nabla f(x_k)^T)^{-1} f(x_k) \tag{6.37}$$

Newton's method defines a sequence of approximations $\{x_0, x_1, x_2, \ldots x_{k-1}, x_k\}$ to the exact solution. When x_0 is chosen to lie "close enough" to the solution, and the function is continuously differentiable, the Newton iteration will converge to the true solution. The following theorem taken from Moré and Sorensen (1984) gives a precise statement of the local convergence properties of Newton's method.[9]

Theorem 6.2. *Let $f : \mathbb{R}^n \to \mathbb{R}^n$ be a continuously differentiable mapping defined on an open set D, and assume that $f(x^*) = 0$ for some x^* in D and that $\nabla f(x^*)^T$ is nonsingular. Then there is an open set S such that for any x_0 in S the Newton iterates (6.37) are well defined, remain in S, and converge to x^*.*

Theorem 6.2 proves that if $x_0 \in S$, the Newton iteration will eventually converge to the solution of the equations x^* as $k \to \infty$. However, in a practical implementation, the iterations are usually terminated once the current iterate is "close enough" to the solution. To decide on when x_k is close enough, we need to know how fast we are progressing toward the solution and how far the current approximation x_k is from the solution. Asymptotic convergence analysis of Newton's method estimates how rapidly the iterates are progressing in the region of the solution, and it provides inequalities that bound the distance from the solution based on the size of the current Newton step. The following definitions of *convergence rate* will be used for the convergence analysis. Define the error e_k of x_k as follows:

$$e_k = \|x_k - x^*\| \tag{6.38}$$

The sequence $\{x_k\}$ is linearly convergent if there exists a constant $\beta \in (0, 1)$ such that

$$e_{k+1} \leq \beta e_k \tag{6.39}$$

for all $k \geq \bar{k}$ where $\bar{k} = \inf\{k \,|\, x_k \in S\}$. However if β is close to unity, this rate may

[9] See Moré and Sorensen (1984) for a proof of this theorem.

not be acceptable. We say the sequence $\{x_k\}$ converges *quadratically* if:

$$e_{k+1} \leq \beta e_k^2 \quad \forall\, k \geq \bar{k} \tag{6.40}$$

The sequence converges *superlinearly* if

$$e_{k+1} \leq \beta_k e_k \quad \forall\, k \geq \bar{k} \tag{6.41}$$

and the sequence $\{\beta_k\}$ converges to zero. Thus, a quadratically convergent series is superlinearly convergent, and a superlinearly convergent series is linearly convergent. Theorem 6.3, also taken from Moré and Sorensen (1984), states the results of the asymptotic convergence analysis for Newton's iteration.[10]

Theorem 6.3. *Let $f : \mathbb{R}^n \to \mathbb{R}^n$ satisfy the assumptions of Theorem 6.2. The sequence $\{x_k\}$ produced by the iteration defined in (6.37) converges superlinearly to x^*. Moreover, if*

$$\left\| \nabla f(x)^T - \nabla f(x^*)^T \right\| \leq \kappa \left\| x - x^* \right\| \tag{6.42}$$

for $x \in D$ and some finite constant $\kappa > 0$, then the sequence converges quadratically to x^*.

Therefore, if x_0 lies within the region of convergence S and f is continuously differentiable at the solution, Newton's method is guaranteed to converge superlinearly. If the termination criterion for the Newton iteration is based on $\|x_{k+1} - x_k\|$, then the asymptotic rate of convergence can be used to bound the distance from the solution. For convenience, define $\Delta x_k \in \mathbb{R}^n$ as the Newton step or update, so we can rewrite (6.37) and the series of iterates that it defines as follows:

$$\begin{aligned} \Delta x_k &= -(\nabla f(x_k)^T)^{-1} f(x_k) && (6.43) \\ x_{k+1} &= x_k + \Delta x_k && (6.44) \end{aligned}$$

[10] See Moré and Sorensen (1984) for the proof.

$$\begin{aligned} &= x_0 + \Delta x_0 + \Delta x_1 + \ldots + \Delta x_k && (6.45) \\ &= x_0 + \sum_{i=0}^{k} \Delta x_i && (6.46) \end{aligned}$$

Since x^* is the limit point of the Newton iterates, (6.46) defines the solution of the system of equations x^* as $k \to \infty$:

$$x^* = x_0 + \sum_{i=0}^{\infty} \Delta x_i \tag{6.47}$$

Hence, (6.46) and (6.47) define the difference between the current iterate and the solution as follows:

$$x_{k+1} - x^* = \sum_{i=k+1}^{\infty} \Delta x_i \tag{6.48}$$

Moreover, since Newton's method is superlinearly convergent, successive iterates satisfy:

$$\|x_{k+1} - x^*\| \leq \beta_k \|x_k - x^*\| \tag{6.49}$$

where β_k satisfies the conditions set forth in (6.41). Therefore, the error in the current Newton iterate can be expressed in terms of the convergence rate by combining (6.49) and (6.48), making use of the triangle inequality:

$$\begin{aligned} \left\| \sum_{i=k+1}^{\infty} \Delta x_i \right\| &\leq \beta_k \left\| \sum_{i=k}^{\infty} \Delta x_i \right\| = \beta_k \left\| \Delta x_k + \sum_{i=k+1}^{\infty} \Delta x_i \right\| && (6.50) \\ \beta_k \left\| \Delta x_k + \sum_{i=k+1}^{\infty} \Delta x_i \right\| &\leq \beta_k \|\Delta x_k\| + \beta_k \left\| \sum_{i=k+1}^{\infty} \Delta x_i \right\| && (6.51) \\ (1 - \beta_k) \left\| \sum_{i=k+1}^{\infty} \Delta x_i \right\| &\leq \beta_k \|\Delta x_k\| && (6.52) \\ \|x_{k+1} - x^*\| &\leq \frac{\beta_k}{1 - \beta_k} \|\Delta x_k\| && (6.53) \end{aligned}$$

Thus, (6.53) shows that the size of the current Newton step ($\|\Delta x_k\|$) provides a bound on the distance from the current iterate x_{k+1} to the solution x^*. For $\beta_k < .5$, the distance to the solution is always less than the norm of the current Newton update. Furthermore, the fact that β_k approaches zero as x_k approaches x^* implies that requiring a small Newton update insures that x_k is very close to x^*. Brenan *et. al.* (1996) estimate the convergence rate whenever two or more corrector iterations have been taken using (6.54).

$$\beta = \left(\frac{\|x_{k+1} - x_k\|}{\|x_1 - x_0\|} \right)^{1/k} \tag{6.54}$$

Since $\{\beta_k\}$ is an absolutely convergent series, the value of β provided by (6.54) overestimates β_k and generates a conservative estimate of the distance from the solution.

Therefore, when exact arithmetic is employed, terminating the Newton iteration based on the norm of the current Newton step provides a rigorous bound on the distance from the final iterate to the exact solution of the system at hand. Given exact arithmetic and a good initial guess, we can determine the appropriate tolerance to achieve any desired accuracy. Furthermore, since the preceding analysis did not specify the norm to be used, any consistent norm can be used when evaluating the termination criteria. For instance, if a bound on the maximum error in any variable is needed, the infinity norm can be used. The choice of norm will in no doubt be affected by the scale of the variables, so either the system should be well-scaled or the norm should be in some way self-correcting. Since the norm employed by BDF integration codes ($\|\cdot\|_{\mathrm{BDF}}$) incorporates the absolute and relative error tolerances specified for each variable, it accounts both for differences in the relative size of the variables and for the fact that the user may wish to calculate some variables more accurately than others.

6.4 Summary

The severe demands and high expectations placed on the numerical solution procedures employed by equation-based modeling environments requires robust and efficient algorithms. This chapter has highlighted the fact that the accuracy of numerical computations is limited by the machine precision, the stability of the numerical algorithm, and the conditioning of the problem. Figure 6-1 provides compelling evidence that sometimes the numerical integration codes may produce inaccurate results without warning. Chapter 7 demonstrates that current BDF integration codes cannot maintain the user requested accuracy when solving some simulations of interest and proves that ill-conditioned corrector iteration matrices can lead to the observed problems. The scaling techniques reviewed in this chapter are extended to sparse unstructured systems to mitigate the effects of ill-conditioning on the systems of interest.

We have also identified the fact that both dynamic optimization and combined discrete/continuous simulation may require the solution of many IVPs during a single simulation or optimization calculation. Thus, the efficiency of the integration codes during the initial phase of integration impacts the solution efficiency more than it does during the solution of continuous dynamic models, which only require the integration to start once. In chapter 8, we introduce a new method to start DAE integration codes efficiently.

Chapter 7

Automatic Scaling of Differential-Algebraic Systems

As argued in section 1.6, detailed modeling of batch processes requires the use of hybrid discrete/continuous simulation applied to differential-algebraic models exhibiting complex and highly nonlinear behavior (Barton, 1994). The advent of sophisticated equation-based discrete/continuous process modeling environments such as ABACUSS (Barton, 1992) ease the burden placed on the modeler by decoupling the model from the solution algorithm, yet they increase the demands and expectations placed on the numerical solution procedures. This problem is further complicated by the fact that during a batch operation state variables may vary over many orders of magnitude (e.g., the composition profile in a batch distillation column or the holdup of the limiting reagent in a batch reaction), and several physical regimes (e.g., the thermodynamic phase changes in a solvent switch operation).

The severe demands placed on the solution procedures are illustrated through the simulation of the batch distillation of wide-boiling azeotropic mixtures.

This chapter demonstrates and explains why the BDF integration techniques are unable to obtain the desired accuracy when simulating such mixtures on desktop workstations. The difficulties are a property of the mathematical model that results in an ill-conditioned corrector iteration matrix during the integration. Note that these problems are not unique to batch distillation, but the batch distillation models provide a convenient system with which to demonstrate the phenomena. In fact, the examples presented clearly demonstrate the previously unreported result that BDF integration codes applied to DAEs are limited by the *accuracy* that can be attained in the corrector iterations. This accuracy is governed by the condition of the corrector iteration matrix, the accuracy to which the iteration matrix and the function residuals have been evaluated, the machine unit roundoff, and the stability of the method used to factor the iteration matrix. We prove that inaccurate solutions of the corrector iteration may be caused by an ill-conditioned corrector matrix.

This chapter also explores scaling techniques to mitigate the problem and identify situations in which these problems can be expected. Since chemical process models give rise to large sparse unstructured corrector iteration matrices, our results will focus on this class of matrices. We have found that these techniques not only improve the accuracy that can be expected, but they can also improve the efficiency of the integration code. The chapter also shows that the problem of ill-conditioning is not necessarily related to stiffness, even for ordinary differential equations in state space form.

7.0.1 Modeling Flexibility Derived from the Automatic Scaling of DAE Models

Automatic scaling of the differential-algebraic models enhances the robustness of the numerical solution procedures. In doing so, it provides additional flexibility to the modeler working within equation-based modeling environments. A common problem when working within commercially available equation-based simulation environments is the need to work within a sometimes inconvenient set of units; for example, SpeedUp (AspenTech, 1993) does not employ SI units in its model libraries. Attempting to use SI units for these same models leads to numerical difficulties. Since the BDF integration codes control both the relative and absolute error (whichever dominates), the numerical difficulties are not the result of a change in the way the error in the solution is measured. Instead, the problems are caused by the conditioning of the linear systems solved during the integration. By automatically scaling the problem during the integration, an equivalent model that is better conditioned is employed during the solution of the linear systems. This renders changes to the units employed during the model development unnecessary, providing the modeler the freedom to work in the units in which he or she is most comfortable. However, the modeler should still ensure that the absolute tolerances for the variables reflect the units specified for those quantities.

7.1 Demonstration of Problem

Batch distillation of wide boiling azeotropic mixtures is common in the specialty chemical and synthetic pharmaceutical industries where a heavy product is separated from volatile solvents and reagents that form azeotropes. The simulation of such operations in ABACUSS provides a dramatic illustration of the limitations imposed by finite precision floating point arithmetic on numerical integration routines. ABACUSS results from the purification of a monomer product from the reagents and solvents employed in its synthesis clearly illustrate the problems that may be encountered.

Although the time profiles of most of the variables are continuous and change smoothly, a handful of variables, such as the condenser duty shown in figure 7-1, appear to contain discontinuities. However, the model has no discontinuities, and the 'spikes' observed are the result of successful integration steps with a very small step size. Figure 7-2 shows that the 'spike' is the result of a successful integration of very small length which supports the fact that the discontinuity checking algorithm (Park and Barton, 1996) reports no events during the simulation. Note that the spikes are not restricted to variables of small magnitude, and the jumps in the variable values are not always in the same direction. In section 7.2, we explain how the BDF code's error control mechanism can permit such behavior, and that the observed behavior can be expected from ill-conditioned systems.

Three index one models[1] of the distillation column were examined to ascertain

[1] The differential index was determined using structural criteria.

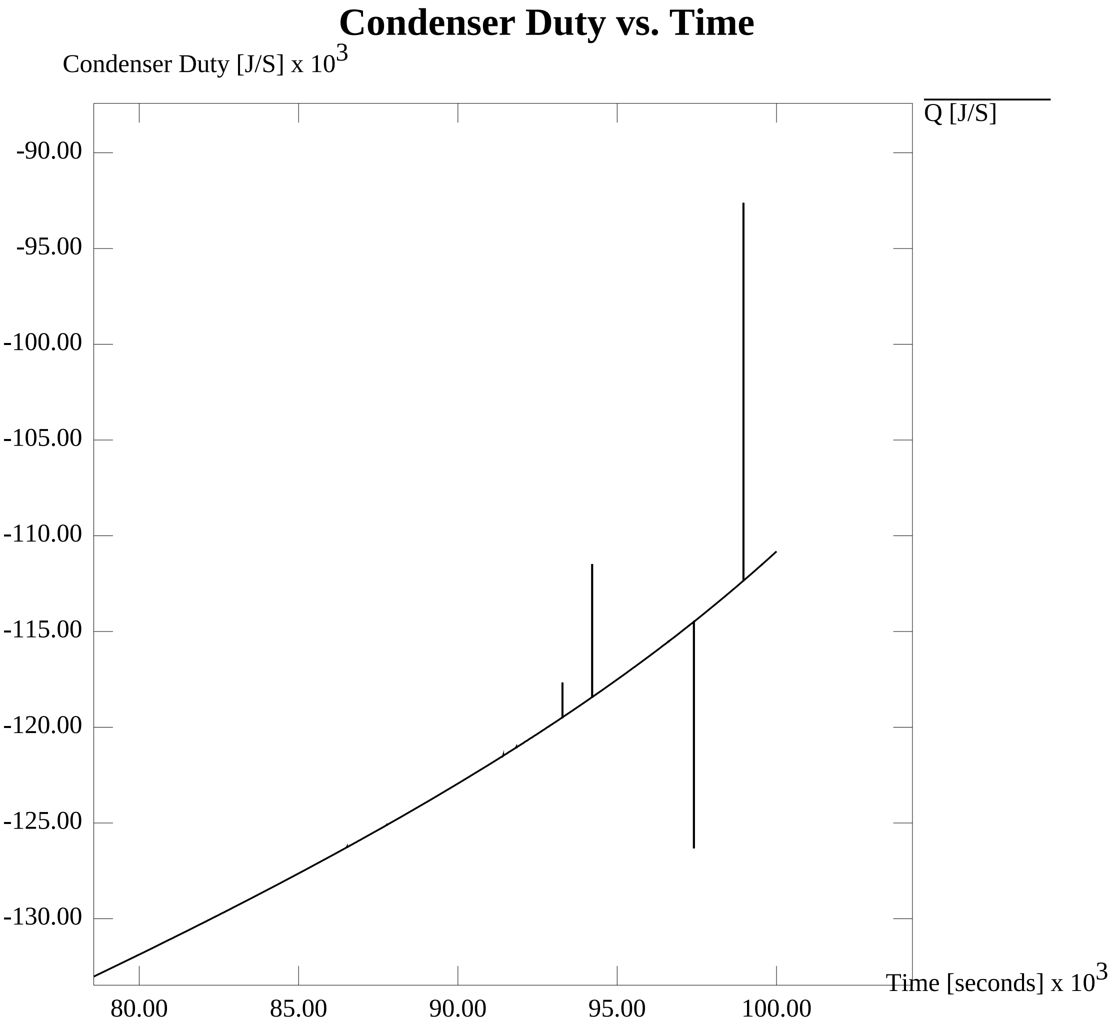

Figure 7-1: “Spikes” in the time profile of the condenser duty.

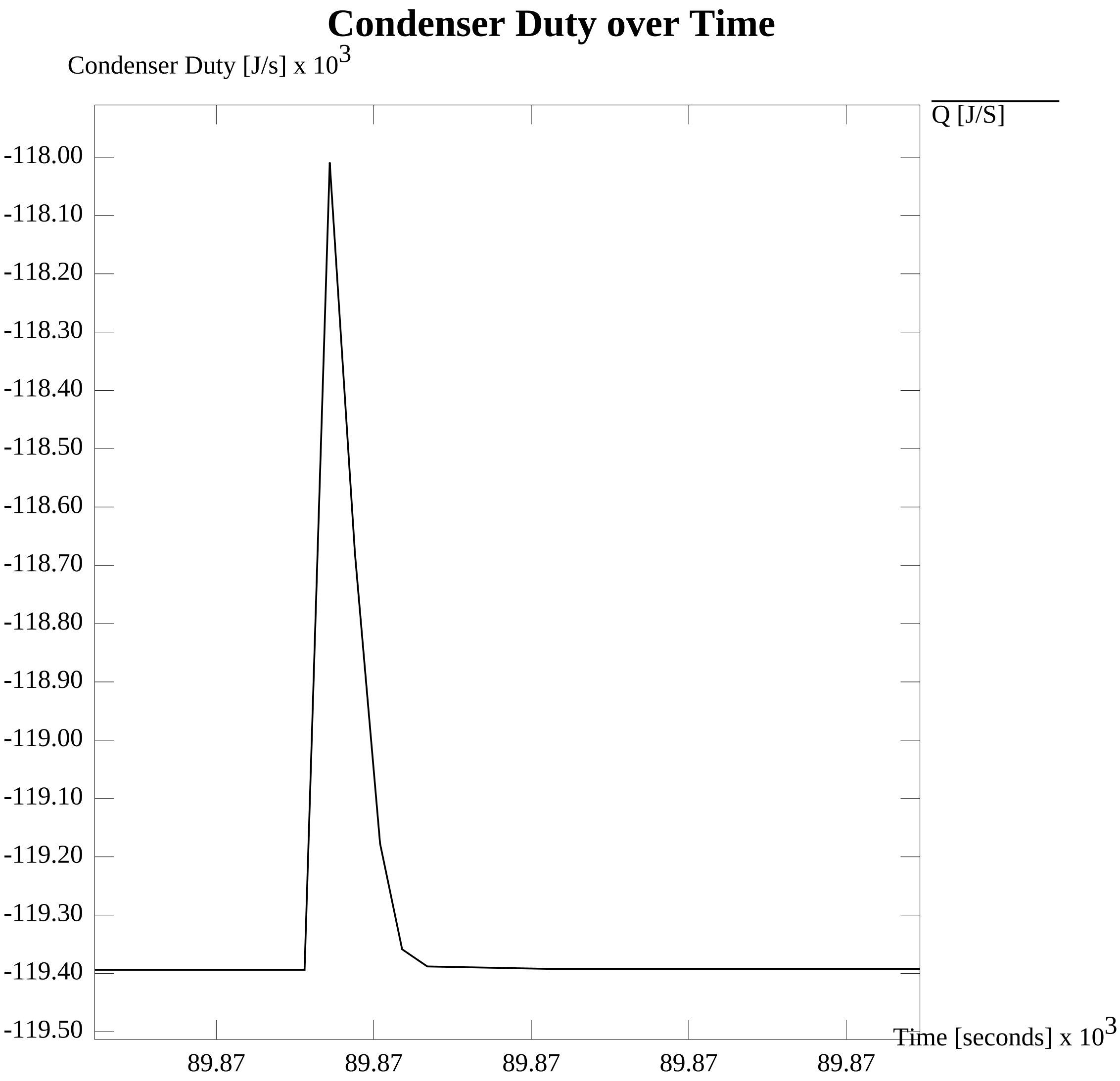

Figure 7-2: One of the ‘spikes’ shown in detail.

whether the numerical difficulties were a property of a particular mathematical abstraction of the physical system, or the underlying physics of the problem. One model contains a static energy balance and constant liquid molar holdup on the trays (Mujtaba and Macchietto, 1991), one approaches the index-2 model used in BatchFrac (Boston *et al.*, 1981), and one relates the vapor and liquid flowrates according to the pressure, tray geometry, and liquid holdup on the trays (Fair *et al.*, 1984; AspenTech, 1995). Simulations performed with each of the three models contained similar spikes whether the liquid phase activity coefficients were modeled using the Wilson equation (Reid *et al.*, 1987) or assumed to be unity. Hence, the phenomenon observed stems from a property of the physical system that is embodied in each of the mathematical abstractions. We infer that the problem is a mathematical property in the resulting systems of equations, and we shall prove this in later sections of this chapter.

We have witnessed this same phenomenon on other models as well. In fact, by constructing models that will lead to ill-conditioned corrector matrices (perhaps ones with an infinite condition number) over a portion of the solution domain, we can expose these numerical problems. For example, consider the following expression approximating the relationship between the flow and the pressure drop across a valve (Jarvis and Pantelides, 1991):

$$f = k_v \sqrt{|P^{in} - P^{out}|}\,\mathrm{sign}(P^{in} - P^{out}) \tag{7.1}$$

where P^{in} and P^{out} represent the upstream and downstream pressures for positive values of the flowrate f. Such an expression leads to an ill-conditioned system when $P^{in} \approx P^{out}$, causing severe numerical problems if flow reversals occur. In fact, when $P^{in} = P^{out}$ no Lipschitz constant for the system exists and, equivalently, the condition number of the Jacobian matrix is infinity. In this case, the undesirable numerical behavior may be averted by making a different modeling approximation (Mandler, 1992):

$$f = \frac{k_v(P^{in} - P^{out})}{b + \sqrt{|P^{in} - P^{out}|}} \tag{7.2}$$

where b is a small positive regularization constant. Experience has clearly shown that the latter modeling approximation performs much better. The former approximation has been shown to lead to the spikes similar to the ones illustrated here on models unrelated to batch distillation. However, if the regularization constant b in (7.2) is made sufficiently small (but not zero), the spiking phenomenon can occur. This demonstrates that spikes may be observed in systems with large, but not infinite, condition numbers; hence, the phenomenon is not restricted to those systems not admitted by the conditions for existence and uniqueness of a solution of an ODE (i.e., those that are not Lipschitz continuous like (7.1)). A model of the Imperial College Pilot Plant (Barton, 1992) was run with using the flow pressure relationship shown in (7.2). For values of the regularization parameter b that were greater than 10^{-6} the model did not produce any spikes. For values below 10^{-7} or when (7.1) was used the model produced a spike that led to the improper determination of a state event.

Two different implementations of the BDF integration method were tested to make sure that the observed problems were not caused by a specific implementation. The first code, DASOLV (Jarvis and Pantelides, 1992), employs a fixed coefficient implementation of the BDF method. It was the application of this code that enabled elucidation of phenomena. We have also used DSL48S for the integration of these models. DSL48S is a version of DASSL (Petzold, 1982a), the widely used fixed leading coefficient BDF code for the solution of DAEs, modified for large sparse unstructured systems.[2] DSL48S did not tend to produce as many spikes as DASOLV on the same models,[3] but it would sometimes fail after the step size became too small. As explained later, failure of the integration code is probably more likely than the appearance of a spike when these situations are encountered. Thus, both codes exhibited similar behavior when integrating these models, so the phenomena are not caused by the implementation of a specific code. In fact, the next section identifies

[2] DSL48S also contains a novel and highly efficient method for the integration of parametric sensitivities (Feehery *et al.*, 1997).

[3] In general, DSL48S is more robust and much more efficient than DASOLV.

the conditioning of the corrector iteration matrix as the source of these numerical difficulties.

7.2 Explanation of the Phenomenon

In this section, we demonstrate that the spikes observed are a numerical artifact introduced by the BDF integration technique, and indicate that a breakdown in the error control strategy has occurred. Solution accuracy is maintained by adapting the step size to control the local truncation error. A step is only accepted after the corrector iteration has converged, meaning that BDF approximation of the model equations (6.4) has been satisfied at t_n, and then after satisfying the truncation error criterion (see figure 6-2). The existence of spikes shows that a solution returned from a converged corrector has managed to pass the truncation error criterion in spite of the fact that the predicted and corrected values differ significantly.

The spikes indicate that the results are inaccurate which severely restricts the application of these results to engineering decisions. Moreover, this phenomenon is extremely detrimental to the efficiency of the integrator, which requires many tiny steps and several Jacobian factorizations before returning to the original trajectory and regaining its previous level of confidence.

Sections 7.2.1–7.3 explain how a spike can be generated. Section 7.2.1 explains the computational sequence of the integration code on the integration step that generates a spike. We then examine how a step can pass the truncation error criterion when the predicted and corrected solutions differ significantly, demonstrating that the truncation error criterion may permit significant changes in some variables over a small integration step. Finally, we examine the cause for the large difference between the predicted and corrected solution. Since the predictor provides a value that is consistent with the past integration steps, it will not indicate an abrupt change from the current trajectory. Section 7.3 demonstrates that the large differences between the corrected and predicted solutions are caused by an ill-conditioned corrector iteration matrix, which permits the converged solution of the corrector iteration to be inaccurate.

7.2.1 Generation of a 'spike'

The spikes are the result of repeated truncation error failures and step reductions. On the first attempt to take the step, the corrector is converged but the truncation error criterion (7.3) is not satisfied. As illustrated by 6-2 the code reduces the step size and attempts the step again. Once again the corrector converges, but the truncation error test is not satisfied. This process continues. After the third error test failure, DASSL reduces the order of the approximation to one and continues the sequence of step reductions. This process of step reductions continues until one of two things occur: the step eventually passes the truncation error test, or the step size becomes smaller than the minimum permitted and the integrator gives up.[4]

The logic behind this procedure is that the local truncation error represents the error from truncating the infinite Taylor series expansion of the solution at t_n after a finite number of terms; the expansion is expressed in terms of backward differences (stored in the code as modified divided differences (Brenan *et al.*, 1996)), so the order of magnitude of the neglected terms is a function of the step size. Thus, the error in the BDF approximation of the solution can be reduced by reducing the step size. In the limit as the step size approaches zero, the truncation error approaches zero.

The truncation error is approximated as a function of the difference between the corrected and predicted solutions at t_n. On the one hand, the solution of the corrector iteration z_n^C solves the kth order BDF approximation of the model equations. On the other hand, the infinite series divided difference approximation of the solution at t_n is exact if the divided differences are defined using $x(t_n)$. The error in the BDF approximation (the local truncation error) is given by the difference between this infinite series and the series containing only $k + 1$ terms. The leading term in the difference between these two series is used to approximate the local truncation error, and it is a multiple of the $k+2$ divided difference, denoted by $\phi_{k+2}(n)$. Since the exact

[4] DASOLV allows eight step reductions before declaring that the step is too small and terminating the integration whereas DASSL permits step reductions until the step becomes too small.

solution at t_n was not determined, $\phi_{k+2}(n)$ is approximated using x_n^C instead of $x(t_n)$; with this approximation, the $\phi_{k+2}(n)$ is equal to the difference between the corrected and predicted solutions at t_n $(x_n^C - x_n^P)$. The coefficient of $\phi_{k+2}(n)$ is a function of the order of the approximation and the past step sizes and defines parameter M appearing in (7.3) whenever the truncation error dominates the interpolation error.

The solution of the corrector iteration x_n^C differs from the exact solution $x(t_n)$ due to error contributions from two sources: the inaccuracy of the BDF approximation of the model equations (the truncation error), and the error from determining the numerical, rather than the exact, solution of (6.4). Following Bujakiewicz (1994), we will refer to the latter error as the algebraic error; we measure the *accuracy* of the corrector iteration in terms of the size of the algebraic error. The algebraic error consists of two contributions, the error from terminating the corrector iteration after a finite number of iterations (termination error) and the error due to the propagation of rounding error during the solution of the linear systems encountered within the corrector iteration (the forward error). The termination error is controlled by the BDF algorithm and is guaranteed to be significantly smaller than the permissible truncation error; the BDF method assumes that the forward error is insignificant. Section 7.3 demonstrates that it is a large forward error, resulting from an ill-conditioned corrector matrix, that leads to inaccurate solutions.

Figure 7-3 shows the values of the predicted and corrected solution at each of the attempted step lengths for a variable that exhibits a spike on this integration step. This figure cannot be used to prove that the corrector solutions are inaccurate, but it certainly provides compelling evidence. The figure shows the converged corrector solution and the predicted solution at each of the step sizes attempted during this integration step; these results were produced by DASOLV. The step was accepted at the eighth attempted step size. The figure illustrates that at the longer attempted step lengths the difference between the predicted and corrected value of this variable was not so large. However, as the step length was reduced, the predicted and corrected solutions diverged. At the largest observed difference between these values, the integration step passed the truncation error criterion. Furthermore, this step was not accepted because $\left|x^C - x^P\right|$ for other system variables was decreasing faster than it was increasing for this variable. In the following section we show why the truncation error permits larger differences between x^C and x^P to be accepted at small step lengths. Section 7.3 explains why the divergence between the corrected and predicted solutions can be expected, since the system becomes more ill-conditioned at smaller step lengths.

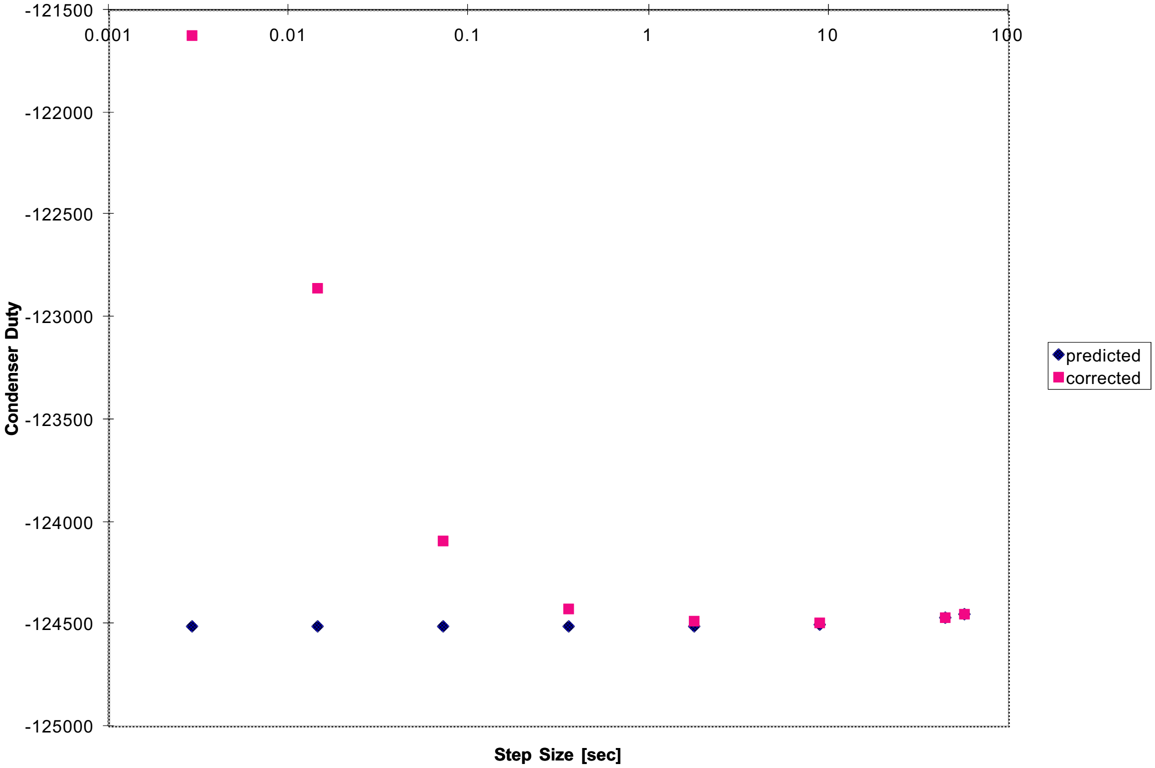

Figure 7-3: A comparison of the predicted and corrected solution as a function of the step size during the generation of a spike.

In this case, the sequence of step reductions generated a spike and permitted the batch distillation simulation to continue; this enabled elucidation of the underlying cause of the problem. However, in many cases the truncation error tolerance is never satisfied and the integration terminates once the step length becomes too small.

7.2.2 Truncation Error Criterion

The step size and order of the BDF approximation is based on the estimates of the accuracy of the BDF approximation provided by the local truncation error. An integration step is only accepted if the local truncation error tolerance is satisfied. The criterion is defined as follows for DASOLV (Jarvis and Pantelides, 1992), DSL48S, and DASSL (Brenan *et al.*, 1996):

$$\text{error} = M \cdot \left\| \mathbf{x}^C - \mathbf{x}^P \right\|_{\text{BDF}} \leq 1.0 \tag{7.3}$$

where $\mathbf{x}^C$ is the corrected solution and $\mathbf{x}^P$ is the predicted solution. Note that the user requested tolerances are buried in the definition of the norm (see (6.7)) used in (7.3). In both DASOLV and the variants of DASSL M varies with the step size (h) and the order of the method. In DASOLV, M is proportional to h. While M is not directly proportional to h in the variants of DASSL, M is proportional to h for a first order method when $h_{n+1} \ll h_n$ as shown below in table 7.1. Therefore, in situations when spikes may be generated, the truncation error scales with the integration step size. With this type of check, if the step size is small enough, almost any value will pass the truncation error check. This is what happens during the creation of the spikes in the example simulations, and either code could accept a step that produces spikes in the values of some variables. Thus, the truncation error check cannot be relied upon to prevent such a spike from being created.

Truncation Error Criteria Imposed by DASSL

DASSL and its variants control both the local truncation error and the interpolation error, the error in the solution at values of t between those at the mesh points t_n. The larger of the two quantities is used to decide whether a step is accepted and to determine the length of the subsequent step. The constant M is defined in terms of the coefficients of the BDF approximation as follows (Brenan *et al.*, 1996):

$$M = \max(\alpha_{k+1}(n+1), |\alpha_{k+1}(n+1) + \alpha_s - \alpha^{\circ}(n+1)|) \tag{7.4}$$

where

$$\alpha_{k+1}(n+1) = \frac{h_{n+1}}{h_{n+1} + h_n + \cdots h_{n+1-k}} \tag{7.5}$$

$$\alpha_s = -\sum_{j=1}^{k} \frac{1}{j} \tag{7.6}$$

$$\alpha^\circ(n+1) = -\frac{h_{n+1}}{\sum_{i=1}^{k} h_{n+2-i}} \tag{7.7}$$

where k represents the current order of the BDF method and n represents the last successful integration step. The first term in the max expression controls the interpolation error and the second controls the local truncation error. M is a function of the current and previous step sizes and the order of the difference approximation. While (7.4–7.7) do not provide much insight of the general behavior of M, two limiting cases are illuminating. Table 7.1 depicts the values of M for first to third order BDF approximations when either the current step length is the same as the previous step (the typical behavior of the code), or when the current step is much smaller than the previous (the behavior that could potentially result in a spike).[5]

BDF	Value of M	
Order	h_i constant	$h_{n+1} \ll h_n$
1	1/2	ϵ
2	1/3	$1/2 - 3\epsilon/2$
3	1/4	$5/6 - 11\epsilon/6$

Table 7.1: Value of the local truncation error parameter M in the limits of constant and drastically reduced step sizes.

The expressions in table 7.1 for the higher order methods assume that the previous steps were roughly the same size (i.e., $h_n = h_{n-1} = h_{n-2}$). We define $\epsilon = h_{n+1}/h_n$ as the ratio of the current to the previous step size, so the terms in the last column are not exact but should be very good approximations. For example, the first term in the last column is $h_{n+1}/(h_{n+1}+h_n)$. The table demonstrates that M is bounded away from

[5] If the current step is much smaller than the previous (e.g., more than three step reductions), then the code switches to a first order method (Brenan *et al.*, 1996).

zero in the higher order methods. Thus, for any of the higher order approximations, DASSL will not accept a step unless the corrected and predicted solutions are close. On the other hand, when the step size is dramatically reduced, DASSL will employ a first order method, so the error control may permit the predicted and corrected solutions to differ by a significant amount since the ϵ can be very small.

7.3 Ill-conditioned Corrector Iterations

Even when the residuals of the equations are evaluated accurately, an ill-conditioned corrector iteration matrix can lead to inaccurate corrector solutions. A set of criteria is derived that defines conditions under which the accuracy of the corrector iteration can be guaranteed in spite of the roundoff error encountered during solution of the Newton updates. The distillation models studied here do not meet these criteria; thus, the corrector iterations admit the possibility of the inaccurate solutions that have been observed in the integration results.

The corrector employs a modified Newton method, terminating iterations when the norm of the numerically calculated update satisfies some tolerance. Assuming that the predictor provides an initial guess within the region of convergence of Newton's method and that the operations are performed using exact arithmetic, the superlinear convergence of Newton's method (Moré and Sorensen, 1984) bounds the distance from the current iterate $\mathbf{x}_k$ to the solution $\mathbf{x}^*$ using the Newton update $\Delta\mathbf{x}$ and the convergence rate β_k according to (7.9). Thus, terminating the Newton iteration when $||\Delta\mathbf{x}||$ satisfies the convergence tolerance τ controls the accuracy of the solution.

$$\begin{aligned} ||\Delta\mathbf{x}|| &\leq \tau && (7.8) \\ ||\mathbf{x}_{k+1} - \mathbf{x}^*|| \leq \frac{\beta_k}{1-\beta_k} ||\Delta\mathbf{x}_k|| &\leq \tau\frac{\beta_k}{1-\beta_k} && (7.9) \end{aligned}$$

Unfortunately, the criterion defined in (7.8) cannot be applied directly because the only information available is the size of the Newton update $\overline{\Delta\mathbf{x}}$ calculated using floating point arithmetic. However, we need only demonstrate that (7.8) is satisfied to assure that the desired accuracy is attained. We employ linear error analysis to derive relationships between $\overline{\Delta\mathbf{x}}$ and the condition number of the iteration matrix $\kappa(\mathbf{J})$ to guarantee that (7.8) holds.

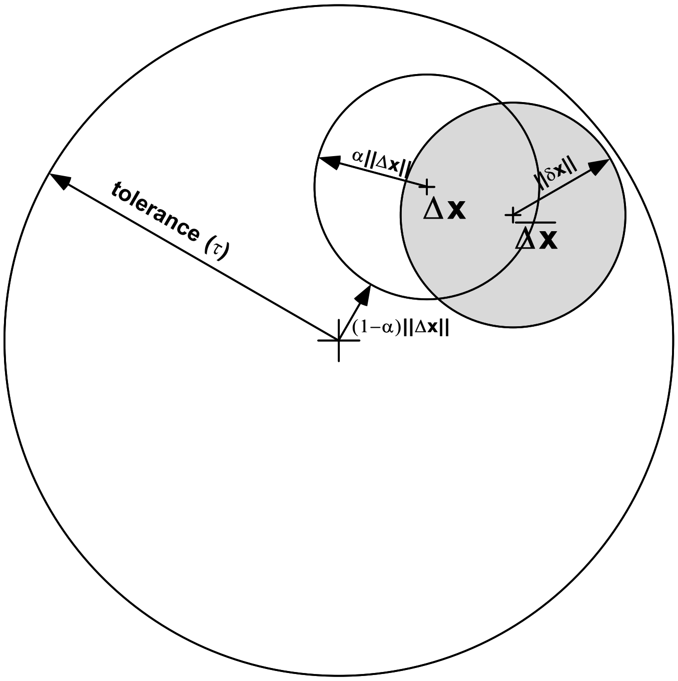

Figure 7-4: Relationship between the exact Newton update $\Delta\mathbf{x}$, the numerically calculated Newton update $\overline{\Delta\mathbf{x}}$, and the convergence tolerance τ.

Criterion (7.8) dictates that $\Delta\mathbf{x}$ must lie in a closed neighborhood of the origin of radius τ, defined by $N_\tau(0)$. Although the exact location of $\Delta\mathbf{x}$ is not known, $\Delta\mathbf{x}$ lies within a closed neighborhood of radius $r = \|\boldsymbol{\delta x}\|$ of the numerically calculated update $\overline{\Delta\mathbf{x}}$. Thus, (7.8) will hold whenever $N_r(\overline{\Delta\mathbf{x}}) \subset N_\tau(0)$. Figure 7-4 illustrates that this condition is satisfied as long as the ball centered around the numerically calculated Newton update is contained within the neighborhood of size τ centered at the origin. The numerical solution of $\mathbf{J}\Delta\mathbf{x} = \mathbf{f}$ is the exact solution of the nearby system $(\mathbf{J} + \boldsymbol{\delta J})(\Delta\mathrm{x} + \boldsymbol{\delta x}) = \mathbf{f} + \boldsymbol{\delta f}$. Using the perturbed system the following bounds are derived for the error in the solution (Duff *et al.*, 1986):

$$\|\boldsymbol{\delta x}\| \leq \frac{\kappa(\mathbf{J})}{\|\mathbf{J}\| - \kappa(\mathbf{J})\,\|\boldsymbol{\delta J}\|}(\|\boldsymbol{\delta f}\| + \|\boldsymbol{\delta J}\|\,\|\Delta\mathrm{x}\|) \tag{7.10}$$

$$\frac{||\boldsymbol{\delta x}||}{||\Delta\mathbf{x}||} \quad \leq \quad \frac{\kappa(\mathbf{J})}{1-\kappa(\mathbf{J})\frac{||\boldsymbol{\delta J}||}{||\mathbf{J}||}}\left(\frac{||\boldsymbol{\delta f}||}{||\mathbf{f}||}+\frac{||\boldsymbol{\delta J}||}{||\mathbf{J}||}\right)=\alpha \tag{7.11}$$

Let (7.11) define α, the bound on the relative error in the Newton update. Define $\Delta\mathbf{x}+\boldsymbol{\delta x}=\overline{\Delta\mathbf{x}}$ and relate the norms.

$$||\Delta\mathbf{x}||-||\boldsymbol{\delta x}|| \leq \left\|\overline{\Delta\mathbf{x}}\right\| \leq ||\Delta\mathbf{x}||+||\boldsymbol{\delta x}|| \tag{7.12}$$

Rearrange (7.12) using the definition of α to produce (7.13) which bounds $||\Delta\mathbf{x}||$ whenever $\alpha < 1$.

$$||\Delta\mathbf{x}|| \leq \frac{\left\|\overline{\Delta\mathbf{x}}\right\|}{1-\alpha} \tag{7.13}$$

Thus, whenever (7.14) is satisfied, then $N_r(\overline{\Delta\mathbf{x}}) \subset N_\tau(0)$, and (7.8) must hold.

$$\left\|\overline{\Delta\mathbf{x}}\right\|/(1-\alpha) \leq \tau \tag{7.14}$$

This demonstrates that for well-conditioned problems with little error in the residual evaluations ($\alpha \to 0$), criterion (7.8) is virtually the same as bounding the numerically calculated update since $\left\|\overline{\Delta\mathbf{x}}\right\| \approx ||\Delta\mathbf{x}||$. However, when the problem is ill-conditioned, $\left\|\overline{\Delta\mathbf{x}}\right\|$ may need to be considerably smaller than $||\Delta\mathbf{x}||$ to ensure that the variables are being controlled to the desired accuracy at the mesh points, indicating that the condition of the iteration matrix should be considered when establishing the convergence criterion that $\left\|\overline{\Delta\mathbf{x}}\right\|$ must satisfy.

If $\alpha \geq 1$, then (7.13) cannot be used to ensure that the accuracy is maintained because $N_r(\Delta\mathbf{x})$ contains the origin. The quantity $\left\|\overline{\Delta\mathbf{x}}\right\|+||\boldsymbol{\delta x}||$ can be overestimated using (7.10) and compared to τ to see if $N_r(\overline{\Delta\mathbf{x}}) \subset N_\tau(0)$; this is discussed in section 7.6. In fact if $||\boldsymbol{\delta x}|| \geq \tau$, then we admit the possibility that the accuracy is not maintained. For ill-conditioned matrices such as the ones encountered in the examples above, we admit this possibility. Even if the residuals are calculated accurately, the calculations are performed without introducing error, and the

Jacobian evaluation is exact, an ill-conditioned corrector can introduce the possibility that the desired accuracy cannot be achieved. For example, consider a Jacobian with $\kappa(\mathbf{J}) = \|\mathbf{J}\| \, \|\mathbf{J}^{-1}\| = 10^5 10^{15} = 10^{20}$. Even if the error $\boldsymbol{\delta J}$ is neglected and the residuals are on the order of 10^{-3} and evaluated to full machine precision[6] ($\|\boldsymbol{\delta f}\| \approx \|\mathbf{f}\| \, u \approx 10^{-19}$), the value of $\|\boldsymbol{\delta x}\|$ could be 10^{-4} according to (7.10), rendering it impossible to guarantee an accuracy of 10^{-5}. This demonstrates that ill-conditioning on its own can admit the possibility of solutions that do not meet the requested accuracy. However, in actual simulations the residuals will not be known this accurately, so the threshold value of $\kappa(\mathbf{J})$ that may lead to problems is reduced. For instance, the rounding error in the difference between two order one variables is on the order of u, roughly 10^{-16}, even if the difference has value 10^{-3}.

This section has demonstrated that the corrector iteration should only be terminated once the desired accuracy has been achieved, not simply when the numerically calculated update has become small. In many cases, these two situations are one in the same, but this is clearly not the case when the iteration matrix is ill-conditioned and the function residuals are not known to full machine precision. In order warn the user of simulations that admit the possibility of introducing errors in excess of the desired accuracy, methods to bound or calculate $\kappa(\mathbf{J})$ and $\|\boldsymbol{\delta f}\|$ are required; efficient methods are needed if these checks are to be performed automatically.

7.4 Stiffness, Conditioning, and Index

It is well known that DAEs represent the limit of an ODE system with infinite transients and that a similar relationship exists between index-1 and higher index DAE systems, etc. In this section, we examine the relationship between the way ODE and DAE systems behave near these limits, and how they behave in the limiting cases (either the DAE or the high index DAE). We demonstrate that ill-conditioning is likely to be a problem near these limits, but that it may be that a well-conditioned solution can be achieved at the limit itself. We demonstrate that the problems which

[6] u represents the machine unit rounding error.

some authors (Chung and Westerberg, 1990; Chung and Westerberg, 1992) have attributed to what they term *near-index* problems are in fact ill-conditioned DAEs. In some cases, ill-conditioned DAEs may occur near the high-index member of a family of models, but this need not be the case.

First, we examine ill-conditioning in terms of the relationship between ODE and DAE systems.

7.4.1 Stiffness and Conditioning of ODEs

In this section, we lay to rest any notion that ill-conditioning of the corrector iteration matrix is simply the result of a model with widely varying time constants. We show that a system may be ill-conditioned when it is not 'stiff', even for constant coefficient linear ODEs in state space form. Since these are merely a subset of DAEs, we can expect that certain DAEs will be ill-conditioned without possessing widely varying time constants.

We examine linear ordinary differential equations in state space form and measure the 'stiffness' according to the stiffness ratio, even though a precise mathematical definition for stiff systems is still argued (Shampine, 1985; Lambert, 1991; Hairer and Wanner, 1991). We consider systems of the following form:

$$\frac{d\mathbf{x}}{dt} = \dot{\mathbf{x}} = \mathbf{A}\mathbf{x} \tag{7.15}$$

where $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{A} \in \mathbb{R}^{n \times n}$. If $\boldsymbol{\lambda} \in \mathbb{R}^n$ defines the eigenvalues of $\mathbf{A}$ ordered such that $|\text{Re}\lambda_1| \geq |\text{Re}\lambda_2| \geq \ldots \geq |\text{Re}\lambda_n|$, the stiffness ratio is defined by $|\text{Re}\lambda_1| \,/\, |\text{Re}\lambda_n|$ (Lambert, 1991). We restrict ourselves to asymptotically stable systems ($\text{Re}\lambda_i < 0$) and demonstrate the following two results: if $\mathbf{A}$ is symmetric, then the condition number of the iteration matrix is always less than the stiffness ratio of the system; if $\mathbf{A}$ is unsymmetric, the corrector iteration matrix can be ill-conditioned even if $|\text{Re}\lambda_1| \,/\, |\text{Re}\lambda_n|$ is an order one quantity.

We define the residual equations and the corrector iteration matrix $\mathbf{J}$ of systems in the form (7.15) as follows:

$$\begin{aligned}
\mathbf{f}(\mathbf{x},\dot{\mathbf{x}}) &= \mathbf{A}\mathbf{x}-\dot{\mathbf{x}} && (7.16)\\
\mathbf{J} &= \left[\frac{\partial \mathbf{f}}{\partial \mathbf{x}}+\frac{\partial \mathbf{f}}{\partial \dot{\mathbf{x}}}\frac{\partial \dot{\mathbf{x}}}{\partial \mathbf{x}}\right] && (7.17)\\
\mathbf{J} &= \left[\mathbf{A}-\frac{\alpha_s}{h}\mathbf{I}\right] && (7.18)
\end{aligned}$$

where α_s is the leading coefficient of the BDF method and h is the integration step size. Let $\tilde{\boldsymbol{\lambda}}$ represent the eigenvalues of $\mathbf{J}$ ordered in the same way as $\boldsymbol{\lambda}$.

Theorem 7.1. *If* $\mathbf{A}$ *is a symmetric matrix, and (7.15) is asymptotically stable, then the condition number of the iteration matrix* $\mathbf{J}$ *is bounded by the stiffness ratio. Specifically, the following holds for any size integration step using* $\boldsymbol{\lambda}$ *and* $\tilde{\boldsymbol{\lambda}}$ *defined above:*

$$1 \le \frac{\left|\tilde{\lambda}_1\right|}{\left|\tilde{\lambda}_n\right|} < \frac{|\lambda_1|}{|\lambda_n|} \quad \forall\ h>0 \tag{7.19}$$

Proof. $\mathbf{J}$ is a symmetric matrix with eigenvalues $\tilde{\lambda}_i$ satisfying $\det(\mathbf{J}-\tilde{\lambda}_i\mathbf{I})=0$. From (7.18) we see that:

$$\det(\mathbf{J}-\tilde{\lambda}_i\mathbf{I}) = \det\left(\mathbf{A}-\left(\frac{\alpha}{h}+\tilde{\lambda}_i\right)\mathbf{I}\right) \tag{7.20}$$

which means that $\alpha/h+\tilde{\lambda}_i$ is an eigenvalue of $\mathbf{A}$. Therefore, the eigenvalues of $\mathbf{J}$ are just shifted by α/h from the corresponding λ_i, so we can define $\tilde{\lambda}_i=\lambda_i-\alpha/h$. We observe that the ratio between the condition number of $\mathbf{J}$, $\tilde{\lambda}_1/\tilde{\lambda}_n$, and the stiffness ratio increases monotonically with h.

$$\frac{d}{dh}\left[\frac{\tilde{\lambda}_1/\tilde{\lambda}_n}{\lambda_1/\lambda_n}\right] = \alpha\frac{\lambda_n}{\lambda_1}\frac{\lambda_n-\lambda_1}{(h\lambda_n-\alpha)^2} > 0 \tag{7.21}$$

Hence, the lower bound on the ratio is defined as $h\to 0$ and the upper bound occurs as $h \to \infty$.

$$\begin{aligned} \lim_{h \to 0} \frac{\tilde{\lambda}_1/\tilde{\lambda}_n}{\lambda_1/\lambda_n} &= \frac{\lambda_n}{\lambda_1} && (7.22) \\ \lim_{h \to \infty} \frac{\tilde{\lambda}_1/\tilde{\lambda}_n}{\lambda_1/\lambda_n} &= 1 && (7.23) \end{aligned}$$

□

However, this bound on the condition number of the iteration matrix does not generalize to arbitrary DAE systems. In fact, it does not even hold for linear time invariant ODEs in state space form if the matrix $\mathbf{A}$ is unsymmetric. Remember that for unsymmetric matrices the condition number is given by the singular values rather than the eigenvalues, and that the singular values and the eigenvalues are unrelated (Strang, 1980). An example demonstrates that the system can have a stiffness ratio near one, but possess an ill-conditioned iteration matrix. Consider the following matrix defined in terms positive real constants a and b.

$$\begin{bmatrix} -1+a & -b \\ 0 & -1-a \end{bmatrix}$$

The eigenvalues of the matrix above lie along the diagonal. By selecting $0 < a \ll 1$, we can see that the stiffness ratio $(1+a)/(1-a)$ remains close to one for any value of b. Selecting b as a large number causes the iteration matrix to become ill-conditioned for a given step size h. However, as we demonstrate in the next section, as the step size h decreases, the corrector iteration matrix becomes better conditioned (in the ODE case).

7.4.2 Conditioning of ODE and DAE systems

Shampine (1993) has noted that ill-conditioning of the corrector matrix does not preclude the accurate solution of systems of ordinary differential equations when BDF methods are used for the integration. He examines the error control procedures and demonstrates that the integration procedure is essentially self-compensating, and the step size control mechanism ensures accurate solution of the equations. However, as the simulation results of section 7.1 have demonstrated, this is not the case for DAE systems. Let's examine why.

The conditioning of the corrector iteration matrix behaves very differently with changes in the step size for ODE and DAE systems. In fact, this is precisely the reason why the situation we have reported cannot occur within ODE systems as Shampine (1993) has demonstrated. Examine the corrector iteration matrix $\mathbf{J}^{\mathrm{ODE}}$ for the ODE system given below:

$$\begin{aligned} \dot{x} &= f(x) && (7.24) \\ \mathbf{J}^{\mathrm{ODE}} &= \frac{h}{\alpha}\frac{\partial f}{\partial x} - \mathbf{I} && (7.25) \end{aligned}$$

and for the DAE system that follows:[7]

$$\begin{aligned} f(\dot{x}, x) &= 0 && (7.26) \\ \mathbf{J}^{\mathrm{DAE}} &= \frac{h}{\alpha_s}\frac{\partial f}{\partial x} + \frac{\partial f}{\partial \dot{x}} && (7.27) \end{aligned}$$

The condition number of these two matrices behave very differently as the step size is reduced. To examine the extreme case, take the limit as the step size tends toward zero:

$$\begin{aligned} \lim_{h \to 0} \kappa(\mathbf{J}^{\mathrm{ODE}}) &= 1 && (7.28) \\ \lim_{h \to 0} \kappa(\mathbf{J}^{\mathrm{DAE}}) &= \infty && (7.29) \end{aligned}$$

since $\partial f / \partial \dot{x}$ is by definition singular for a DAE (Petzold, 1982b).

Now consider the behavior of each of these two systems when a truncation error failure is encountered. In either system, the truncation error failure triggers a step reduction, which improves the accuracy of the predicted solution. For the ODE case, the step reduction improves the condition of the corrector iteration matrix, which in

[7] Note that this matrix differs by a factor of h/α from the form of the corrector iteration matrix that is usually presented, but this does not change the condition number of the matrix.

turn improves the accuracy of the solution to the corrector; therefore, the predicted and corrected solutions will eventually converge. On the other hand, the step size reduction increases the condition of the corrector iteration matrix of the DAE system. If the original truncation error failure was do to an inaccurate predictor, then the step reduction may permit the smaller step to be accepted. On the other hand, if the step originally failed because the corrector solution was inaccurate, reducing the step size will tend to make the situation worse. The predicted and corrected solutions will diverge as illustrated in figure 7-3, causing another truncation error failure and the cycle will continue. Typically, this will result in several step reductions until the step size reaches the minimum allowable length; the integrator will then quit. In rare situations, the fact that the truncation error scales with the step length may permit the step to be accepted after repeated step reductions, in spite of the fact that the difference between the predicted and corrected values of some variables may be large. This results in the spikes that we have observed.

The situation is even more dramatic if a standard BDF integration code is applied directly to a high index DAE. In this case, the condition number of the corrector iteration matrix scales as $(1/h)^m$ where m is the index of the DAE (Brenan *et al.*, 1996). Bujakiewicz (1994) shows that the positive powers of $(1/h)^{m-1}$ appearing in the matrix inverse cause an amplification of the truncation error by corresponding powers of $1/h$. In fact, this is precisely the reason why standard BDF integration codes often fail when applied to high index problems, in spite of the fact that the truncation error breaks down and solution accuracy cannot be maintained even if the integration continues (Petzold, 1982b). Any truncation error failure triggers a step size reduction which tends to amplify the truncation error due to the increased error in the corrector; eventually the step size becomes so small the integrator gives up.

7.4.3 Modeling Decisions Related to the Index

Modeling assumptions can be made that are equivalent to taking the asymptotic limit of another model. This is one way to view the relationship between DAEs and ODEs. It is well known that DAEs represent the limit of an ODE system as the stiffness ratio tends toward infinity (Brenan *et al.*, 1996). Consider:

$$y' = f(y, z, \epsilon) \tag{7.30}$$

$$\epsilon z' = g(y, z, \epsilon) \tag{7.31}$$

where ϵ is a small number, making the system stiff. When $\epsilon = 0$, the following DAE system is obtained.

$$y' = f(y, z) \tag{7.32}$$

$$0 = g(y, z) \tag{7.33}$$

The stiffness ratio of the DAE system is infinite if we employ the ODE definition of stiffness, but we have removed the fast transient from the problem and required that the solution lie on a lower dimensional manifold defined by the DAE (i.e., satisfying (7.33)). Observe that the components of the solution of the ODE not lying on the DAE solution manifold rapidly decay away (see Hairer *et al.* (1993)) for small ϵ.

A similar relationship exists between some index-1 and high index DAEs; the following system serves as an example:

$$\dot{x}_1 = -x_1 - y \tag{7.34}$$

$$\dot{x}_2 = -x_2 - y \tag{7.35}$$

$$x_1 - \epsilon y = \sin(t) \tag{7.36}$$

When $\epsilon = 0$, (7.34–7.36) form an index-2 DAE, and for $\epsilon \neq 0$ the DAE is index-1. As ϵ approaches zero, the solution of the index-1 DAE approaches the solution of the high index system; components of the solution not lying on the solution manifold of the index-2 system rapidly decay away. Figure 7-5 shows the values of x_1 and x_2 versus time for $\epsilon = 10^{-3}$ and the index-2 problem, demonstrating that the solution is close to that of the high index system. In fact, the solutions lie on top of each other. Figures 7-6 and 7-7 show how the value of y at the start of the simulation decays onto the high index manifold for various values of ϵ

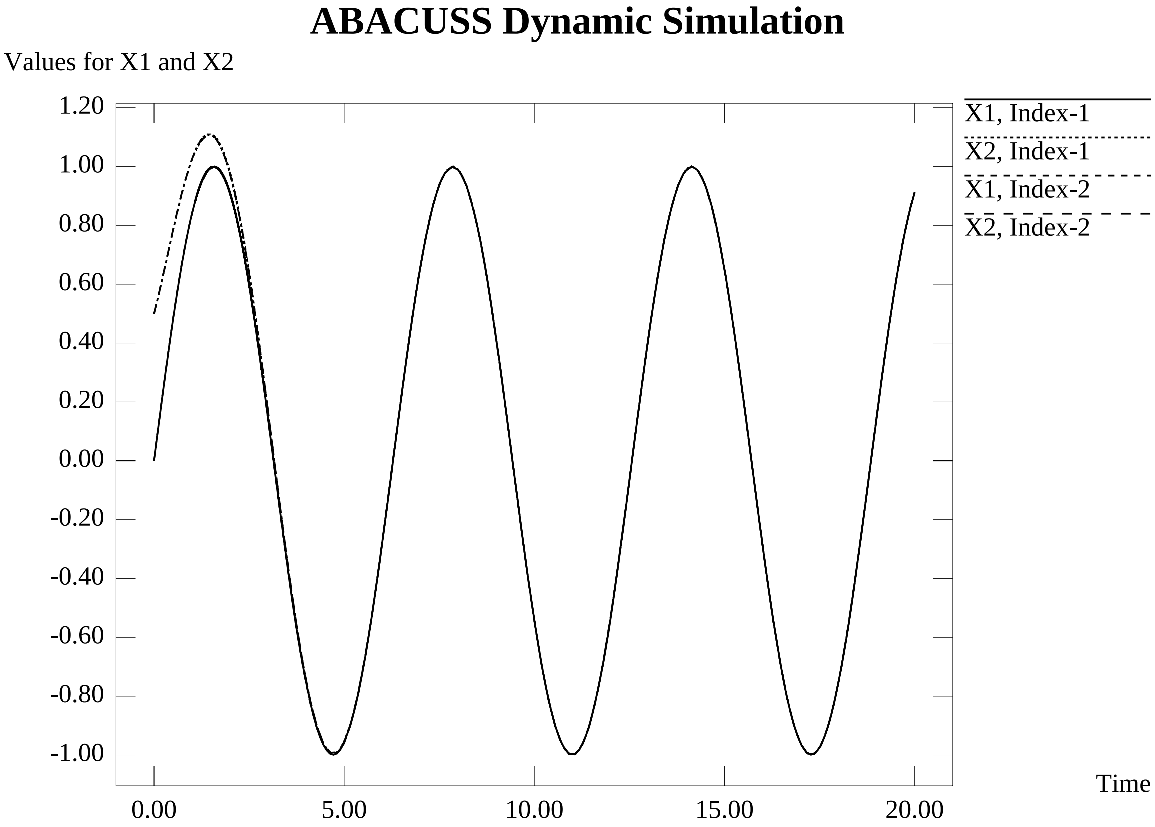

Figure 7-5: Values for x_1 and x_2 for the index-2 system and when $\epsilon = 10^{-3}$.

Does it make sense to solve the high index system instead of the index-1 system? First, we determine whether the difference between the solution of the high index system and that obtained for nonzero values of ϵ is small enough to be ignored during the application of the results. If not, there is no point in proceeding further. If the difference is small enough, then we compare whether the high index model is easier to solve. The high index model can be solved by automatically transforming the high index system to an equivalent index-1 DAE using the method of dummy derivatives (Mattsson and Söderlind, 1993) implemented within ABACUSS (Feehery and Barton, 1995); the method is demonstrated in the next section. Note that the equivalent index-1 system contains more equations. Table 7.2 shows that the high index model is substantially easier to solve than the index-1 model for small values

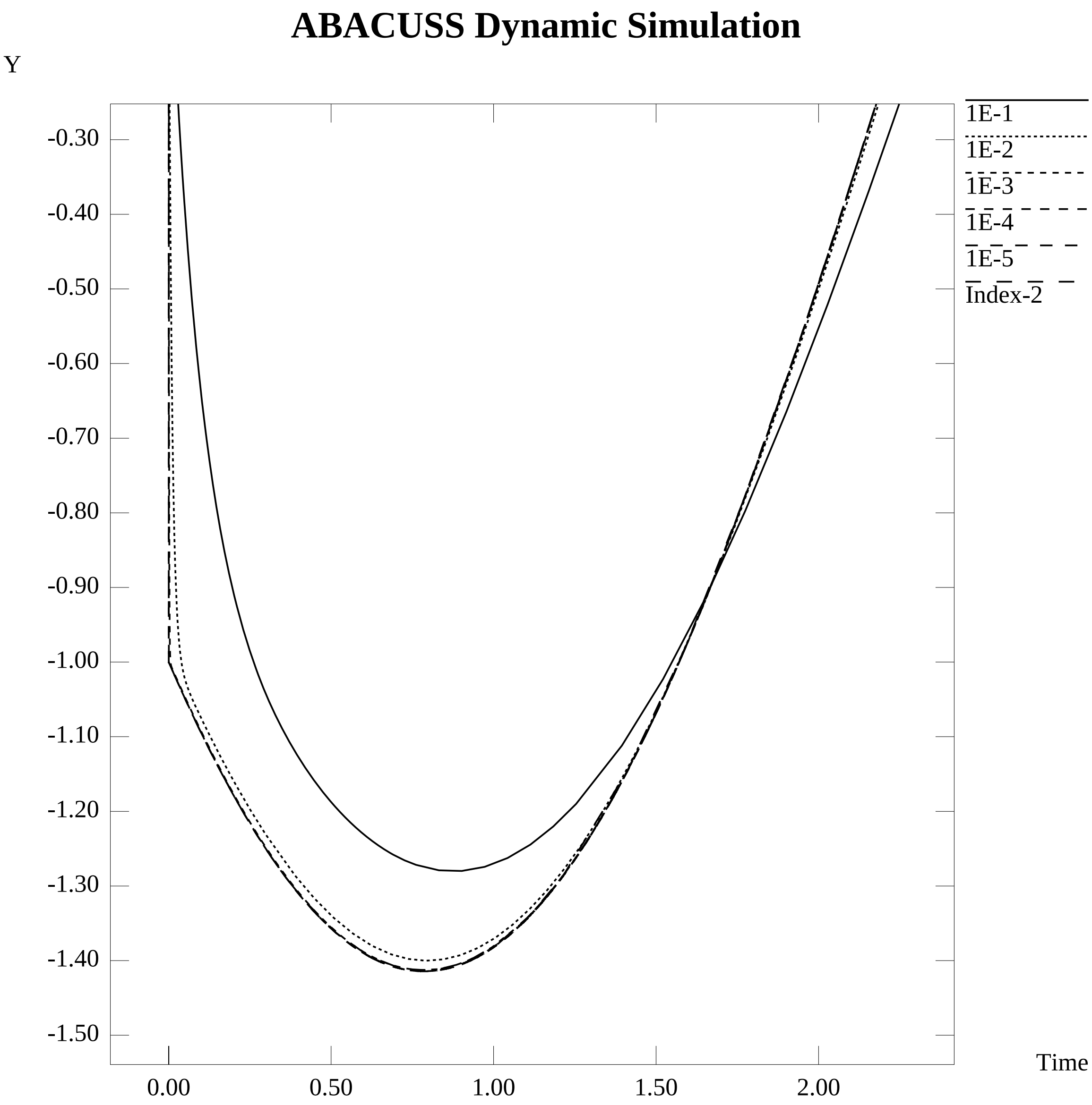

Figure 7-6: Demonstration of the difference between $\epsilon = .1$ and the other values of ϵ.

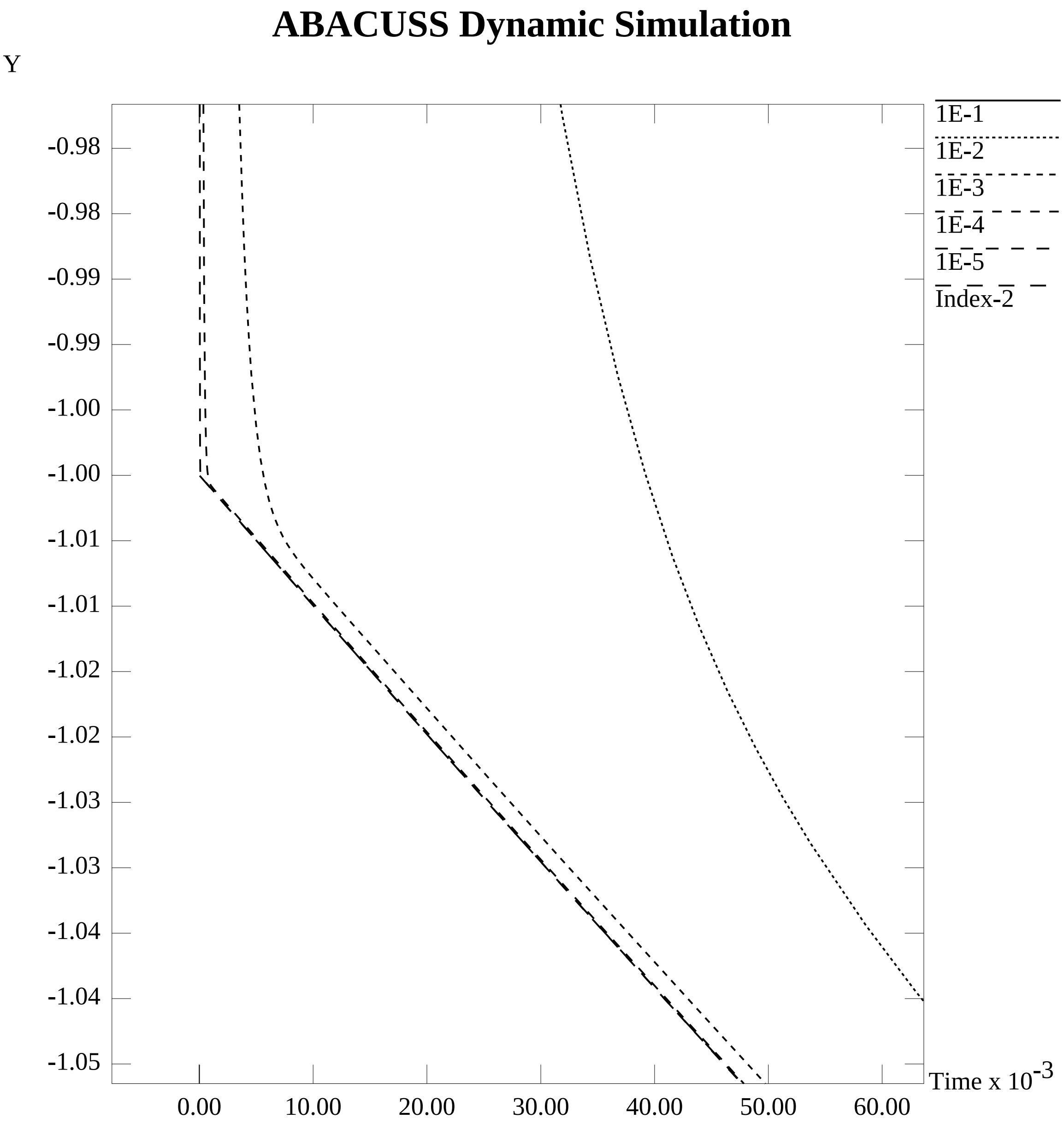

Figure 7-7: The decay of y onto the high index manifold for different ϵ.

ϵ	Jacobian Factorizations	Integration Steps	Residual Evaluations	Convergence Failures	Error Failures
1×10^{-1}	14	210	422	0	1
1×10^{-2}	25	238	501	0	7
1×10^{-3}	9501	5040	19395	0	4744
1×10^{-4}	136848	68704	228001	0	68404
1×10^{-5}	61213	43646	152498	0	30594
0	12	136	273	0	1

Table 7.2: Numerical statistics for the solution of (7.34–7.36) at different values of ϵ.

of ϵ.[8]

This example demonstrates that in some cases it may be beneficial to make modeling assumptions that require the solution of the high index DAE because the numerical solution of the equivalent index-1 system obtained using the method of dummy derivatives is better behaved that the original index-1 system that was approaching the high index problem.

7.4.4 The myth of 'Near Index' Systems

As section 7.4.3 demonstrated, we can make modeling decisions that lead to a higher index problem (e.g., an index-1 or high index DAE) in which the solution lies in a space of reduced dimensionality. This limits the degrees of freedom with which to specify the initial condition because the initial condition must lie within the reduced space. What modeling assumptions are made is simply a modeling decision that should be based on the validity of the approximation, although they may also impact the efficiency of the solution procedure as shown above. In some cases, these modeling assumptions are not valid, so we cannot hope to introduce a method to transform systems automatically. To illustrate this point, let's examine the solution technique for 'near index' problems studied by Chung and Westerberg (1990; 1992). Their examples clearly show the danger of such a procedure, and indicate that the behavior of the high index system may be qualitatively different from that of the lower index

[8] The statistics presented are for the DSL48S integrator embedded within ABACUSS. DASOLV failed to produce a solution for all values of ϵ below .001.

system as it parametrically approaches the high index system.

Chung and Westerberg (1992) consider the following DAE:

$$\begin{aligned} f_1(x,\dot{x},y,t) &= \dot{x}_1 - x_2 = 0 && (7.37) \\ f_2(x,\dot{x},y,t) &= \dot{x}_2 - y = 0 && (7.38) \\ f_3(x,\dot{x},y,t) &= x_1 - \epsilon y - g(t) = 0 && (7.39) \end{aligned}$$

When $\epsilon = 0$, (7.37–7.39) form an index-3 DAE and for $\epsilon \neq 0$ the system is index-1.

We employ the method of Mattsson and Söderlind (1993) to derive an equivalent index-1 model corresponding to the index-3 system, such as the following system:

$$\begin{aligned} x_1' - x_2' &= 0 && (7.40) \\ x_2' - y &= 0 && (7.41) \\ x_1 &= g(t) && (7.42) \\ x_1' &= \frac{\partial g}{\partial t} && (7.43) \\ x_2' &= \frac{\partial^2 g}{\partial t^2} && (7.44) \end{aligned}$$

where the variables x_1' and x_2' are the dummy derivatives that have been introduced. Observe that this system contains no degrees of freedom with which to specify the initial condition and amounts to an analytic solution to the problem. All variables in the system are algebraically related to the forcing function $g(t)$. Selecting $g(t) = \sin(t)$ (following Chung and Westerberg (1992)), we obtain the solution shown in figure 7-8 in which all of the variables are defined in terms of sine and cosine functions and vary over the range [-1,1].

Thus, the solution of the high index system is bounded and is easy to obtain. Now we examine the solution of the index-1 system for $g(t) = \sin(t)$. To ease the derivation of the analytic solution, eliminate the algebraic variable y from (7.37–7.39) to yield the following ODE:

$$\dot{x}_1 = x_2 \qquad (7.45)$$

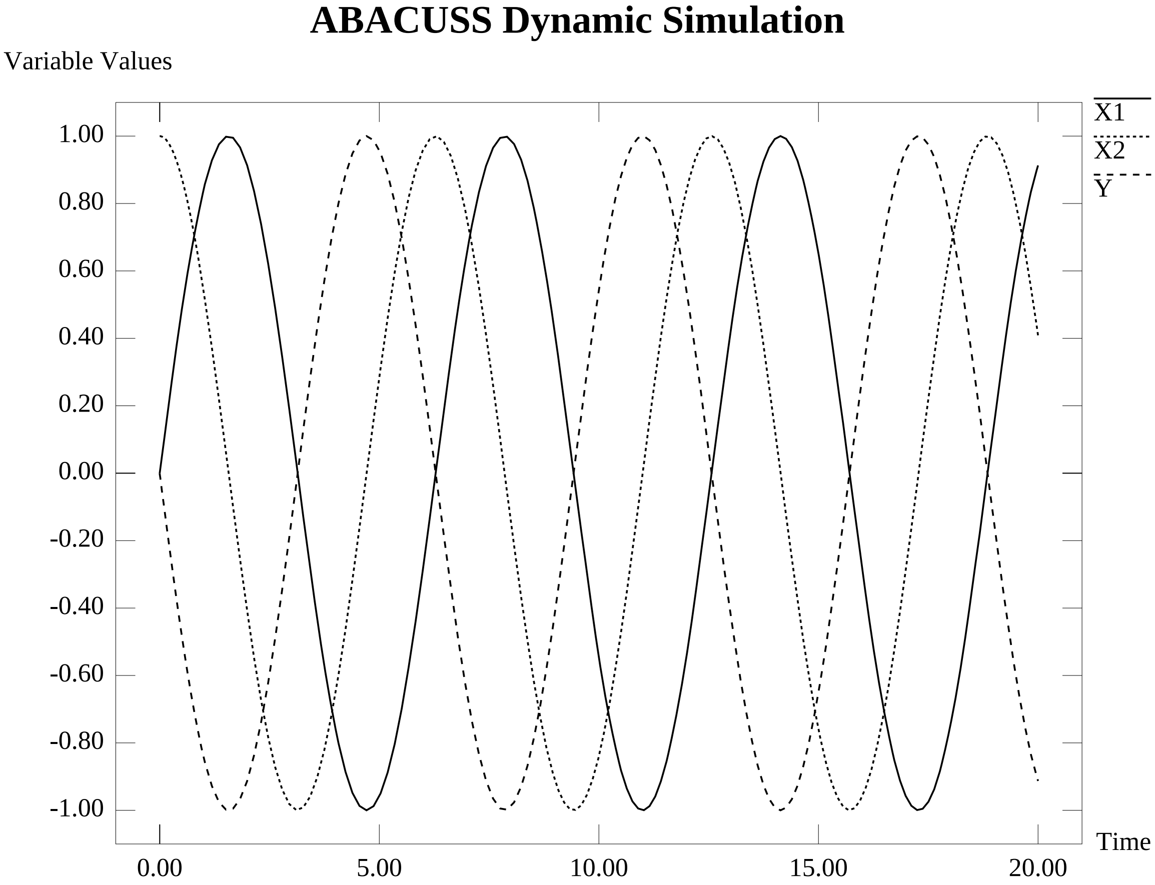

Figure 7-8: The solution the index-3 system found by solving the equivalent index-1 system (7.40–7.44).

$$\dot{x}_2 \quad = \quad \frac{1}{\epsilon}\left(x + \sin(t)\right) \tag{7.46}$$

The general solution of the linear constant coefficient ODE (7.45–7.46) is given below in terms of the parameter ϵ:

$$x_1(t) \quad = \quad \frac{\sin(t)}{1+\epsilon} + C_1 e^{t/\sqrt{\epsilon}} + C_2 e^{-t/\sqrt{\epsilon}} \tag{7.47}$$
$$x_2(t) \quad = \quad \frac{\cos(t)}{1+\epsilon} + \frac{1}{\sqrt{\epsilon}}\left(C_1 e^{t/\sqrt{\epsilon}} - C_2 e^{-t/\sqrt{\epsilon}}\right) \tag{7.48}$$

where the constants C_1 and C_2 are determined by the initial condition. Note that this system is unstable; any rounding error in the initial condition or introduced during the integration procedure will grow exponentially. Although the analytic solution remains bounded for the special case in which the initial condition specified requires that $C_1 = 0$,[9] any attempt to integrate this system numerically will result in a solution that grows exponentially since perturbations to the initial condition are introduced by rounding error and these will grow in an unbounded fashion.

Integrating the index-1 system within ABACUSS demonstrates the fact that the system is unstable. Values of ϵ approaching zero simply make the solution grow more rapidly. Figure 7-9 shows the solution for $\epsilon = .5$, $x_1(0) = 0$, $x_2(0) = 1$. The initial values of x_1 and x_2 place the solution on the manifold defined by the high index system at the initial time.

The algorithm proposed by Chung and Westerberg (1992) calculates the solution of (7.37–7.39) as a perturbation of the high index solution. A perturbation of the high index system cannot capture the qualitative behavior of the index-1 system (i.e., instability). Their results define a bounded oscillating solution for the index one model for small values of ϵ; their algorithm has stabilized the unstable system onto the solution manifold defined by the high index system. Clearly, the solution of the nearby high index system does not behave the same way as the index-1 model does as the limit is approached, since the index-1 DAE does not decay onto the solution manifold defined by the high index DAE. Therefore, the modeling approximation

[9] For positive ϵ. $C_2 = 0$ would lead to a stable analytic solution for $\epsilon < 0$.

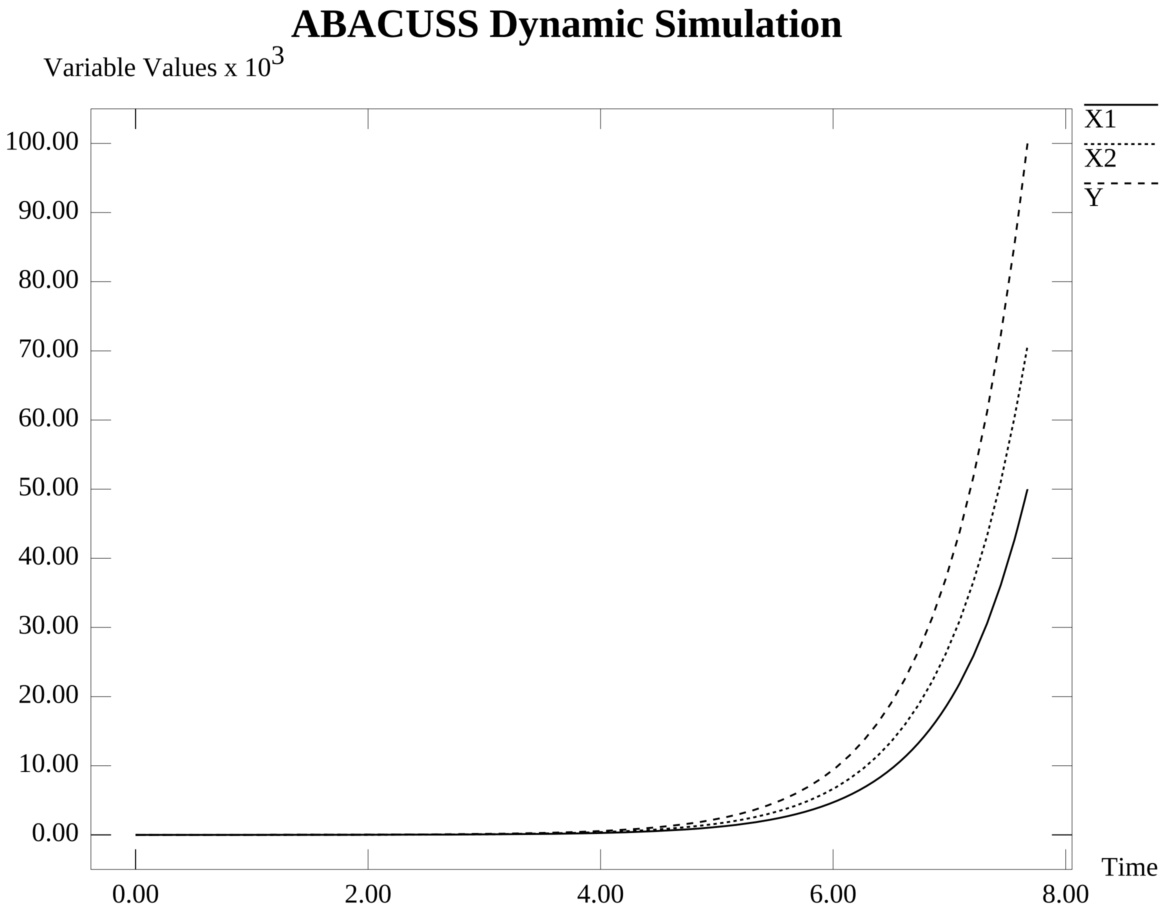

Figure 7-9: The unstable solution of the index-1 system.

setting $\epsilon = 0$ is not valid and should not be made. This example highlights the danger of blindly transforming an index-1 system to the 'nearby' high index system, indicating that 'near index' systems do not, in general, exist. Some of the arguments employed in the Chung and Westerberg paper (1992) to demonstrate the existence of near index systems were mathematically incorrect, so the authors were obviously led to incorrect conclusions. In addition, they applied their algorithm to several unstable systems, but never mentioned or recognized that the systems were unstable. However, in some cases, such as those demonstrated in section 7.4.3, the behavior of the high index system represents the limit of the index-1 DAE and the modeler may choose to formulate the high index system to improve the solution efficiency.

7.5 Scaling Variables and Equations

Scaling the linear system solved at each corrector iteration offers the potential to increase the accuracy of the solution obtained. Typical scaling methods (reviewed in section 6.3.4) employ two diagonal scaling matrices to transform the original system (7.49) into a scaled equivalent (7.50). The choice of scaling matrices encompasses two issues: the condition of the scaled system and the validity of measuring the error in the scaled system of variables. If the condition number of the scaled system is considerably smaller than the original, then we expect a more accurate answer in terms of the transformed variables $\Delta\mathbf{y} = \mathbf{D}_2^{-1}\Delta\mathbf{x}$ (Golub and Van Loan, 1989).

$$\begin{aligned} \mathbf{J}\Delta\mathbf{x} &= \mathbf{f} && (7.49) \\ (\mathbf{D}_1\mathbf{J}\mathbf{D}_2)\Delta\mathbf{y} &= \mathbf{D}_1\mathbf{f} && (7.50) \end{aligned}$$

However, accuracy can only be improved if the scaling can be performed without introducing any significant error. As long as the diagonal elements of the scaling matrices are restricted to integer powers of the machine base, the transformation is exact even if it is performed using finite precision floating point arithmetic. The mantissas are not altered, so no rounding error is introduced (ANSI/IEEE Std. 754, 1985).

Diagonal matrices that minimize the condition number of the scaled system exist (Braatz and Morari, 1994), yet their determination requires $\mathbf{J}^{-1}$, so calculating them is clearly not an option when our goal is to improve the accuracy of the solution to (7.49) in an efficient manner. We have implemented a scaling strategy to improve the accuracy of $\Delta\mathbf{x}$, measured in the norm used by the integrator, at each corrector iteration. The strategy employs column scaling followed by row equilibration using diagonal matrices composed of elements that are integer powers of the machine base. When the error is measured in the norm used by the integrator, this scaling policy brings the condition number of the scaled system close to the minimum value that can be achieved using any diagonal matrices. This scaling policy improves the bounds on the relative solution error. The details of the row and column scaling algorithms employed are justified and explained in the following sections.

7.5.1 Scaling the Variables

The way in which the error is measured dictates the choice of the matrix $\mathbf{D}_2$ used to scale the variables. The matrix $\mathbf{D}_2$ could be chosen to minimize the condition of the column scaled equivalent $\mathbf{J}_C = \mathbf{J}\mathbf{D}_2$, but as van der Sluis (1970) has shown $\kappa(\mathbf{J}\mathbf{D}_2)$ may provide misleading information about the accuracy in the solution of (7.49) if the way in which the error in $\Delta\mathbf{x}$ is measured is important. In fact, he states that selecting $\mathbf{D}_2$ to minimize $\kappa(\mathbf{J}\mathbf{D}_2)$ is similar to answering the question "in which norm does the error *look* most favorable" (van der Sluis, 1970). Since we would like the condition number of the resulting system to be indicative of the quality of the solution that will be obtained, we select $\mathbf{D}_2$ to reflect our error criterion.

The default norm used by the BDF integration routines to estimate the truncation error and measure the size of the corrector updates was defined in (6.7) and has been repeated here for convenience:

$$\|\mathbf{x}\|_{\mathrm{BDF}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left|\frac{x_i}{\tau_{ri}\,|x_i^p| + \tau_{ai}}\right|^2}$$

where x_i^p is the value of the variable x_i from the previous integration step, τ_{ri} is the relative error tolerance and τ_{ai} is the absolute error tolerance for variable i. This weighted root mean square norm is equivalent to $1/\sqrt{n}$ times the Euclidean norm in the transformed system of coordinates, $\mathbf{D}_2^{-1}\Delta\mathbf{x}$, when $\mathbf{D}_2$ is chosen according to (7.51).

$$\mathbf{D}_2 = \{\mathbf{D} \in \mathbf{R}^{n\times n} \;\;:\;\; d_{ii} = \tau_{ri}\,|x_i^p| + \tau_{ai}, \quad d_{ij} = 0 \;\; \forall\, i \neq j\} \tag{7.51}$$

The condition number of the transformed matrix $\mathbf{J}_C$ provides an indication of the quality of the solution that can be expected from the solution of the linear system (7.49) in the absence of row scaling. Let $\boldsymbol{\delta x}$ and $\boldsymbol{\delta f}$ represent the error in $\mathbf{f}$ and $\Delta\mathbf{x}$ respectively. Assuming that the only error introduced during the calculation is due to the initial storage of $\mathbf{f}$, then linear error analysis shows that $||\boldsymbol{\delta x}||\,/\,||\Delta\mathbf{x}|| \;\leq\; \kappa(\mathbf{J})\,||\boldsymbol{\delta f}||\,/\,||\mathbf{f}||$. However, the quality of the solution of (7.49) is given by $||\boldsymbol{\delta x}||_{\mathrm{BDF}}\,/\,||\Delta\mathbf{x}||_{\mathrm{BDF}}$. A bound on this quantity is provided by the same linear error analysis applied to the transformed system shown in (7.52).

$$\begin{aligned}
\mathbf{J}\mathbf{D}_2\mathbf{D}_2^{-1}\Delta\mathbf{x} &= \mathbf{J}_C\Delta\mathbf{y} = \mathbf{f} && (7.52)\\
\frac{||\boldsymbol{\delta x}||_{\mathrm{BDF}}}{||\Delta\mathbf{x}||_{\mathrm{BDF}}} = \frac{||\boldsymbol{\delta y}||_2}{||\Delta\mathbf{y}||_2} &\leq \kappa_2(\mathbf{J}_C)\frac{||\boldsymbol{\delta f}||_2}{||\mathbf{f}||_2} && (7.53)
\end{aligned}$$

Thus, when $\mathbf{D}_2$ is selected according to (7.51), $\kappa_2(\mathbf{J}_C)$ is the condition number that reflects the accuracy of the solution of the linear system.

Scaling the variables in this way is easy to implement and has several advantages. It reflects the physics of the problem by using information that is available within the integrator and passes this information to the linear algebra. It permits the modeler to work in a convenient set of units, greatly diminishing the need to select units for the simulation variables merely to improve the performance of the numerical algorithms.[10] It automatically adapts when variables change over many orders of magnitude during the course of the simulation, a common occurrence in batch process simulations.

[10] The consistent initialization of such problems is not affected by this scaling and remains sensitive to the units selected.

In addition, this scaling ensures that the magnitude of each of the elements of the iteration matrix properly reflects the way in which the error of the linear system will be measured; the selection of pivots during Gaussian elimination and the selection of the scale factors used during row equilibration are governed by the magnitude of the elements in the iteration matrix. Since pivots are selected to reduce the growth in the solution error and the row scaling factors are chosen to reduce the condition of the linear system, choosing $\mathbf{D}_2$ to reflect the way in which the integrator measures the error should result in a more accurate solution of the linear system in terms of the BDF norm. Furthermore, the condition of the scaled iteration matrix can be calculated using a Euclidean norm; this provides the condition of the original matrix calculated according to the norm used by the integrator. Therefore, the condition of the scaled iteration matrix indicates the difficulty in obtaining an accurate solution in terms of the way in which the integrator measures accuracy. Using the scaled iteration matrix $\mathbf{J}_C$, the accuracy criterion (7.14) derived in section 7.3 can be applied using the condition number defined on the two norm.

To implement the scaling defined above as part of a numerical algorithm, $\mathbf{D}_2$ is approximated using integer powers of the machine base β. This provides the matrix $\hat{\mathbf{D}}_2$ defined in (7.54) for a base two machine.

$$\hat{\mathbf{D}}_2 = \{\mathbf{D} \in \mathbf{R}^{n\times n} \quad : \quad d_{ii} = 2^{\left\lfloor \log_2\left(\tau_{r\,i}\left|x_i^p\right| + \tau_{a\,i}\right)\right\rfloor}, \quad d_{ij} = 0 \;\; \forall \; i \neq j\} \tag{7.54}$$

Only integer powers of the machine base need to be stored to define the matrix. These can be calculated efficiently using the functions recommended in the IEEE floating point standard (ANSI/IEEE Std. 754, 1985).

7.5.2 Scaling the Equations

The equations are scaled to minimize the condition number of the column scaled iteration matrix. The scaling employed balances the rows of $\mathbf{J}_C$; an integer scale factor is chosen so that the scaled norm of each row is between one and β.

$$\mathbf{D}_1 = \{\mathbf{D} \in \mathbf{R}^{n\times n} \quad : \quad d_{ii} = \beta^{-\left\lfloor \log_\beta(\|\mathbf{J}_i\|)\right\rfloor}, \quad d_{ij} = 0 \ \ \forall\, i \neq j\} \tag{7.55}$$

In the rest of this section we will demonstrate that although this approach does not guarantee a reduction of the condition of the matrix, for the sparse matrices in which we are interested, it guarantees that the condition of the scaled matrix is close to the condition of the optimally row scaled matrix. We extend the results of van der Sluis (1969) to prove that the scaling matrix defined in (7.55) provides a β-scaled equivalent of $\mathbf{J}_C$ with a condition number that is within a factor of $\beta\sqrt{q}$ of the optimally row scaled matrix defined on the two norm, where q is the maximum number of non-zero elements in any column of $\mathbf{J}_C$.

Van der Sluis (1969) generalized the work of Bauer (1963), proving the row equilibration theorem and demonstrating that *row equilibration* can satisfy the optimal row scaling for a fairly wide class of norms. However, row equilibration does not find the optimal scaling matrix to minimize the condition number defined on the two norm, which is the condition number of $\mathbf{J}_C$ that reflects the fact that the error is measured in the BDF norm. We extend this work to show that simple row equilibration allows us to determine a β-scaled equivalent of the iteration matrix that is with a factor of $\beta\sqrt{q}$ of the optimal.

Van der Sluis (1969) used the row equilibration theorem (theorem 6.1) to show that $\kappa_2(\tilde{\mathbf{D}}\mathbf{A})$ is within a factor of $\sqrt{m}$ of the optimally scaled matrix in terms of the two norm. We extend his result (6.36) to sparse matrices in the following corollary.

Corollary 7.1. *Let $\tilde{\mathbf{D}}$ be the scaling matrix that equilibrates the two norm of the rows of $\tilde{\mathbf{D}}\mathbf{A}$. The condition number of $\tilde{\mathbf{D}}\mathbf{A}$ defined on the two norm is within a factor of $\sqrt{q}$ of the condition number of the optimal row scaled matrix, so:*

$$\frac{\kappa_2(\tilde{\mathbf{D}}\mathbf{A})}{\min_{\mathbf{D}\in\mathcal{D}_m} \kappa_2(\mathbf{D}\mathbf{A})} \le \sqrt{q} \tag{7.56}$$

Proof. Given $\|\mathbf{A}\|_2 \le \sqrt{q}\max_j \left\|(\mathbf{A}^H)_j\right\|_2$ (van der Sluis, 1969), we obtain the following inequality.

$$\frac{\left\|\tilde{\mathbf{D}}\mathbf{A}\right\|_2}{\phi(\tilde{\mathbf{D}}\mathbf{A})} \leq \sqrt{q}\frac{\max_j \left\|((\tilde{\mathbf{D}}\mathbf{A})^H)_j\right\|_2}{\phi(\tilde{\mathbf{D}}\mathbf{A})} \tag{7.57}$$

We employ the fact that $\tilde{\mathbf{D}}\mathbf{A}$ is row equilibrated and divide both sides of (7.57).

$$\frac{\frac{\left\|\tilde{\mathbf{D}}\mathbf{A}\right\|_2}{\phi(\tilde{\mathbf{D}}\mathbf{A})}}{\min_{\mathbf{D}\in\mathcal{D}_m}\frac{\|\mathbf{D}\mathbf{A}\|_2}{\phi(\mathbf{D}\mathbf{A})}} \leq \frac{\sqrt{q}\frac{\left\|((\tilde{\mathbf{D}}\mathbf{A})^H)_k\right\|_2}{\phi(\tilde{\mathbf{D}}\mathbf{A})}}{\min_{\mathbf{D}\in\mathcal{D}_m}\frac{\|\mathbf{D}\mathbf{A}\|_2}{\phi(\mathbf{D}\mathbf{A})}} \quad \forall\ k = 1, 2, \ldots m \tag{7.58}$$

Use (6.34) and (6.35) to substitute for the numerator on the right hand side of (7.58):

$$\frac{\frac{\left\|\tilde{\mathbf{D}}\mathbf{A}\right\|_2}{\phi(\tilde{\mathbf{D}}\mathbf{A})}}{\min_{\mathbf{D}\in\mathcal{D}_m}\frac{\|\mathbf{D}\mathbf{A}\|_2}{\phi(\mathbf{D}\mathbf{A})}} \leq \sqrt{q}\frac{\min_{\mathbf{D}\in\mathcal{D}_m}\frac{\|\mathbf{D}\mathbf{A}\|_2}{\phi(\mathbf{D}\mathbf{A})}}{\min_{\mathbf{D}\in\mathcal{D}_m}\frac{\|\mathbf{D}\mathbf{A}\|_2}{\phi(\mathbf{D}\mathbf{A})}} \quad \forall\ k = 1, 2, \ldots m \tag{7.59}$$

which simplifies to the desired result for the appropriate choice of ϕ:

$$\frac{\frac{\left\|\tilde{\mathbf{D}}\mathbf{A}\right\|_2}{\phi(\tilde{\mathbf{D}}\mathbf{A})}}{\min_{\mathbf{D}\in\mathcal{D}_m}\frac{\|\mathbf{D}\mathbf{A}\|_2}{\phi(\mathbf{D}\mathbf{A})}} \leq \sqrt{q} \tag{7.60}$$

Let $\phi(\mathbf{A}) = \inf_{\mathbf{x}\in\mathbb{R}^n, \|\mathbf{x}\|\neq 0} \|\mathbf{A}\mathbf{x}\|_2 / \|\mathbf{x}\|_2 = \|\mathbf{A}^{-1}\|_2$. □

Row-equilibration "solves" the scaling problem for certain classes of norms and bounds the distance to the optimal for $\kappa_2(\mathbf{D}\mathbf{A})$ when optimizing over $\mathbf{D} \in \mathcal{D}_n$. However, when using the matrix in a numerical algorithm, the scaling matrix must be selected from the space of diagonal matrices consisting of integer powers of the machine base to eliminate the possibility of introducing roundoff error during the transformation.[11] At first glance, this indicates that an integer programming problem must be solved to find the optimal scaling matrix, but theorem 7.2 demonstrates that a solution with condition number that is within a factor β of the best obtainable can be found easily.

[11] An added benefit is that functions usually exist (ANSI/IEEE Std. 754, 1985) to manipulate the exponent of the floating point number directly, allowing such manipulations to be performed extremely efficiently.

Let $\hat{\mathcal{D}}_m$ be the class of nonsingular $m \times m$ diagonal matrices with nonzero elements that are integer powers of the machine base ($\hat{d}_{ii} = \beta^{\eta_i}$ where η_i is any integer). The following theorem proves that the optimal value of $\psi(\mathbf{DB})/\phi(\mathbf{DA})$ over $\hat{\mathcal{D}}_m$ is within a factor of β of the optimum over $\mathcal{D}_m$ for any functions ψ and ϕ satisfying the assumptions of the row equilibration theorem.

Theorem 7.2. *For $\mathbf{A}, \mathbf{B} \in \mathbf{R}^{m \times n}$ with $\psi(\mathbf{DB}) = \max_j \left\|((\mathbf{DB})^H)_j\right\|_\gamma$, where $\|\cdot\|_\gamma$ is an absolute norm, and $\phi(\mathbf{DA})$ is left-monotonic on $\mathcal{D}_m\mathbf{A}$, define $\tilde{\mathbf{D}} \in \mathcal{D}_m$ as the matrix that minimizes $\psi(\mathbf{DB})/\phi(\mathbf{DA})$ over $\mathcal{D}_m$. Let $\eta_i = -\left\lfloor \log_\beta(\tilde{d}_{ii}) \right\rfloor$ and define $\hat{\mathbf{D}}$ as an integer approximation to $\tilde{\mathbf{D}}$ with elements $\hat{d}_{ii}$ defined as follows*

$$\hat{d}_{ii} = \begin{cases} \beta^{\eta_i} & \text{if} \quad \beta^{\eta_i} \leq \tilde{d}_{ii} < \beta^{\eta_i+1-\omega} \\ \beta^{\eta_i+1} & \text{if} \quad \beta^{\eta_i+1-\omega} \leq \tilde{d}_{ii} < \beta^{\eta_i+1} \end{cases} \tag{7.61}$$

The minimum of $\psi(\mathbf{DB})/\phi(\mathbf{DA})$ over $\hat{\mathcal{D}}_n$ is within a factor of β of the optimal over $\mathcal{D}_m$, so

$$\min_{\mathbf{D} \in \mathcal{D}_m} \frac{\psi(\mathbf{DB})}{\phi(\mathbf{DA})} \leq \min_{\mathbf{D} \in \hat{\mathcal{D}}_m} \frac{\psi(\mathbf{DB})}{\phi(\mathbf{DA})} < \beta \min_{\mathbf{D} \in \mathcal{D}_m} \frac{\psi(\mathbf{DB})}{\phi(\mathbf{DA})} \tag{7.62}$$

Furthermore, rounding the nonzero elements of $\tilde{\mathbf{D}}$ to an integer power of the machine base according to (7.61) defines the matrix $\hat{\mathbf{D}} \in \hat{\mathcal{D}}_m$ that satisfies (7.63) for all ω such that $0 \leq \omega < 1$.[12]

$$\frac{\psi(\hat{\mathbf{D}}\mathbf{B})}{\phi(\hat{\mathbf{D}}\mathbf{A})} < \beta \min_{\mathbf{D} \in \mathcal{D}_m} \frac{\psi(\mathbf{DB})}{\phi(\mathbf{DA})} \tag{7.63}$$

Proof. Let $\check{\mathcal{D}}_m^1$ and $\check{\mathcal{D}}_m^2$ be the classes of nonsingular $m \times m$ diagonal matrices with nonzero elements satisfying $\check{d}_{ii}^1 \in (1/\beta, 1]$ and $\check{d}_{ii}^2 \in [1, \beta)$ for $i = 1 \dots m$ respectively.

[12] The parameter ω allows the theorem to apply whether the elements of the diagonal scaling matrix $\tilde{\mathbf{D}}$ is rounded up or down to the nearest integer power of two.

Define $\check{\mathcal{D}}_m = \check{\mathcal{D}}^1_m \cup \check{\mathcal{D}}^2_m$. Since $\check{\mathcal{D}}^k_m \hat{\mathcal{D}}_m = \mathcal{D}_m$ for $k = 1, 2$,

$$\min_{\mathbf{D}\in\mathcal{D}_m} \frac{\psi(\mathbf{DB})}{\phi(\mathbf{DA})} = \min_{\mathbf{D}^f\in\check{\mathcal{D}}^k_m, \mathbf{D}^\beta\in\hat{\mathcal{D}}_m} \frac{\psi(\mathbf{D}^f\mathbf{D}^\beta\mathbf{B})}{\phi(\mathbf{D}^f\mathbf{D}^\beta\mathbf{A})} \tag{7.64}$$

Using the facts that ϕ is left-monotonic and $\max_j \left\|((\mathbf{DB})^T)_j\right\|_\gamma \geq \min_k |d_{kk}| \max_j \left\|(\mathbf{B}^T)_j\right\|_\gamma$, we have the following.

$$\begin{aligned}
\min_{\mathbf{D}^f\in\check{\mathcal{D}}^k_m, \mathbf{D}^\beta\in\hat{\mathcal{D}}_m} \frac{\psi(\mathbf{D}^f\mathbf{D}^\beta\mathbf{B})}{\phi(\mathbf{D}^f\mathbf{D}^\beta\mathbf{A})} &\geq \min_{\mathbf{D}^f\in\check{\mathcal{D}}^k_m, \mathbf{D}^\beta\in\hat{\mathcal{D}}_m} \frac{\psi(\mathbf{D}^f\mathbf{D}^\beta\mathbf{B})}{\max_j |d^f_{jj}|\phi(\mathbf{D}^\beta\mathbf{A})} && (7.65)\\
&\geq \min_{\mathbf{D}^\beta\in\hat{\mathcal{D}}_m} \frac{\min_i |d^f_{ii}|\psi(\mathbf{D}^\beta\mathbf{B})}{\max_j |d^f_{jj}|\phi(\mathbf{D}^\beta\mathbf{A})} && (7.66)\\
&> \min_{\mathbf{D}^\beta\in\hat{\mathcal{D}}_m} \frac{1}{\beta}\frac{\psi(\mathbf{D}^\beta\mathbf{B})}{\phi(\mathbf{D}^\beta\mathbf{A})} && (7.67)
\end{aligned}$$

The left hand inequality in (7.62) is self evident.

To prove the second part of the theorem, define $\check{\mathbf{D}} = \hat{\mathbf{D}}^{-1}\tilde{\mathbf{D}}$ and use (7.61) to show the following holds:

$$\begin{array}{rclcrcl}
1 \leq & \check{d}_{ii} & \leq \beta^{1-\omega} & \text{if} & \beta^{\eta_i} \leq & \tilde{d}_{ii} & \leq \beta^{\eta_i+1-\omega} \\
\beta^{-\omega} < & \check{d}_{ii} & < 1 & \text{if} & \beta^{\eta_i+1-\omega} < & \tilde{d}_{ii} & < \beta^{\eta_i+1}
\end{array} \tag{7.68}$$

Therefore, $\check{d}_{jj} \in (\beta^{-1}, \beta^{1-\omega})$ for all $j = 1, \ldots m$. Substitute $\hat{\mathbf{D}}\check{\mathbf{D}}$ for $\tilde{\mathbf{D}}$ to show that $\hat{\mathbf{D}}$ satisfies the same bound given for the optimal $\mathbf{D} \in \hat{\mathcal{D}}_m$.

$$\min_{\mathbf{D}\in\mathcal{D}_m} \frac{\psi(\mathbf{DB})}{\phi(\mathbf{DA})} = \frac{\psi(\check{\mathbf{D}}\hat{\mathbf{D}}\mathbf{B})}{\phi(\check{\mathbf{D}}\hat{\mathbf{D}}\mathbf{A})} \geq \frac{\min_i |\check{d}_{ii}|\psi(\hat{\mathbf{D}}\mathbf{B})}{\max_j |\check{d}_{jj}|\phi(\hat{\mathbf{D}}\mathbf{A})} > \frac{\beta^{-\omega}\psi(\hat{\mathbf{D}}\mathbf{B})}{\beta^{1-\omega}\phi(\hat{\mathbf{D}}\mathbf{A})} = \frac{1}{\beta}\frac{\psi(\hat{\mathbf{D}}\mathbf{B})}{\phi(\hat{\mathbf{D}}\mathbf{A})} \tag{7.69}$$

□

Theorem 7.2 demonstrates that we can easily determine a row scaling that generates a β-scaled equivalent of $\mathbf{A}$ that is within a factor of β of the best possible row scaled matrix for any condition number for which the row equilibration theorem applies. Moreover, if $\mathbf{A} = \mathbf{B} \in \mathbf{R}^{n\times n}$ and $\tilde{\mathbf{D}}$ equilibrates the two norm of the rows of $\mathbf{A}$ and $\phi(\mathbf{A}) = \inf_{\mathbf{x}\in\mathbb{R}^n, \|\mathbf{x}\|\neq 0} \|\mathbf{Ax}\|_2 / \|\mathbf{x}\|_2$, then combining the results of (7.57) and (7.62) demonstrates that $\kappa_2(\hat{\mathbf{D}}\mathbf{A})$ is within a factor of $\beta\sqrt{q}$ from the minimum

$\kappa_2(\mathbf{DA})$ over $\mathbf{D} \in \mathcal{D}_n$ for the diagonal matrix $\hat{\mathbf{D}}$ defined in (7.61).

$$\frac{\left\|\hat{\mathbf{D}}\mathbf{A}\right\|_2}{\phi(\hat{\mathbf{D}}\mathbf{A})} \leq \sqrt{q}\frac{\max_j \left\|((\hat{\mathbf{D}}\mathbf{A})^T)_j\right\|_2}{\phi(\hat{\mathbf{D}}\mathbf{A})}$$
$$< \beta\sqrt{q} \min_{\mathbf{D}\in\mathcal{D}_m} \frac{\max_j \left\|((\mathbf{DA})^T)_j\right\|_2}{\phi(\mathbf{DA})} \leq \beta\sqrt{q} \min_{\mathbf{D}\in\mathcal{D}_m} \frac{\|\mathbf{DA}\|_2}{\phi(\mathbf{DA})} \quad (7.70)$$

Equation (7.70) shows that we can easily calculate a β-scaled equivalent of the scaled corrector iteration matrix $\mathbf{J}_C$, simply by equilibrating the two norm of the rows of $\mathbf{J}_C$ and rounding to an integer power of the machine base according to (7.61). Thus, we can scale the corrector iteration matrix by first scaling its columns, and then scale its rows using the matrix $\hat{\mathbf{D}}$. The resulting matrix is within a factor $\beta\sqrt{q}$ of the optimally scaled matrix in terms of the way the integrator measures the error, and it can be calculated efficiently.

The scaling algorithm that we have implemented within DASOLV and DSL48S scales the iteration matrix every time it is reevaluated, typically after several integration steps. First the columns of the matrix are scaled, then the rows are scaled using the $\hat{\mathbf{D}}$ defined in (7.61) when $\omega = 0$. However, the η_i used to define $\hat{\mathbf{D}}$ are determined without actually calculating the two norms of the rows of $\mathbf{J}_C$ to make the algorithm more efficient. Let $\mathbf{j}_i = (\mathbf{J}_C^T)_i$. The η_i must satisfy the definition given in theorem 7.2 which can be rearranged as follows to eliminate the square root operation (proved in theorem A.4):

$$\eta_i = -\left\lfloor \log_\beta(\|\mathbf{j}_i\|_2)\right\rfloor = -\left\lfloor \log_\beta\left(\left(\mathbf{j}_i^T\mathbf{j}_i\right)^{1/2}\right)\right\rfloor = -\left\lfloor \frac{1}{2}\log_\beta\left(\mathbf{j}_i^T\mathbf{j}_i\right)\right\rfloor \quad (7.71)$$

Since the rows of $\mathbf{J}_C$ are sparse, the dot product $\mathbf{j}_i^T\mathbf{j}_i$ can be calculated efficiently (requiring $O(\tau/n)$ operations[13]). The floor of $\log_\beta$ can be performed very efficiently by employing the logb function recommended by the IEEE floating point standard (ANSI/IEEE Std. 754, 1985) which is defined as $\mathsf{logb}(x) = \left\lfloor\left|\log_\beta(x)\right|\right\rfloor$. The func-

[13] Duff *et al.* (1986) define τ as the number of nonzero entries in the matrix, so τ/n is the average number of nonzeroes per row.

tion merely accesses the exponent of the binary representation of the floating point number. The right hand side of (7.71) can be rewritten using the fact that $\lfloor y/k \rfloor = \lfloor \lfloor y \rfloor /k \rfloor$ for any integer k to make use of logb. Thus, η_i can be calculated according to (7.72); the cost of this calculation is dominated by the computation of the dot product of a sparse vector.

$$\eta_i = \left\lfloor \frac{1}{2} \mathsf{logb}(\mathbf{j}_i^T \mathbf{j}_i) \right\rfloor \tag{7.72}$$

Only the vector $\boldsymbol{\eta}$ needs to be stored to represent the matrix $\hat{\mathbf{D}}$. The iteration matrix may be scaled either implicitly (Forsythe and Moler, 1967) during the calculation or explicitly before it is factored using the function scalb (ANSI/IEEE Std. 754, 1985). Currently, we use explicit scaling to avoid making any changes to the sparse linear algebra routines.

The combination of row and column scaling just presented has reduced the condition number of the iteration matrix on all of the example problems on which it has been tested. In fact, scaling has reduced the condition number on the batch distillation models from approximately 10^{22} to 10^8. Thus, the scaling method offers the opportunity to extend the range of problems that may be solved with a machine of given precision. If the condition number of the corrector iteration matrix is reduced substantially, the scaled iteration matrix may meet the conditions under which accuracy may be guaranteed while the unscaled matrix cannot. The scaling has entirely eliminated the spikes observed on some simulations where the requested tolerance has been loosened, and on other simulations the scaling has reduced the number of spikes that have been observed. The scaled simulation is solved more efficiently because there are fewer truncation error failures, residual evaluations, and Jacobian factorizations. However, in both cases the residual error combined with the condition of the iteration matrix are not sufficient to guarantee the accuracy of the solution.

An efficient implementation of this scaling has been coded directly within the DASOLV and DSL48S integrators and is transparent to the user.[14] The scaling algo-

[14] The user merely sets a flag to indicate that scaling should be performed. In ABACUSS, this

rithm can be applied to any BDF integration code on any machine that adheres to the IEEE floating point standard, although it is most suited for sparse systems. We have already ported and tested the scaling on more than five Unix platforms. Preliminary results show improvements to the integrator's efficiency even on problems that can be solved accurately with the standard integration scheme; these improvements are more noticeable using DASOLV.

7.6 Automatic Detection of Potential Inaccuracy

The criterion indicating an accurate solution of the corrector iteration (7.14) can be checked after the solution of every corrector iteration. In order to apply this criterion the following quantities are required:

1. A bound on the error in the residual values.
2. The condition number of the corrector iteration matrix, or an upper bound on the condition number.
3. A bound on the backward error resulting from the LU factorization of the corrector iteration matrix.

In order to apply the criterion automatically, these quantities must be calculated efficiently so the checks can be performed without reducing the efficiency of the numerical integration code.

A bound on the rounding error in the equation residuals can be calculated every time the terms in the corrector iteration matrix are evaluated provided that the reverse mode of automatic differentiation is employed (Iri *et al.*, 1988; Iri and Kubota, 1991; Kubota and Iri, 1991). Since the reverse mode of automatic differentiation is implemented within ABACUSS (Tolsma and Barton, 1997), estimates of the residual error $\boldsymbol{\delta f}$ are available. In addition, the backward error resulting from the LU factorization of the iteration matrix can be determined during the factorization of the matrix itself at little additional cost (Reid, 1987; Arioli *et al.*, 1989).

option can be set at run time.

In general, estimating the condition number of a matrix is a difficult task. However, since the matrix with which we are dealing is a sparse, row-equilibrated matrix, fairly tight upper bounds on the condition number can be obtained. For a row equilibrated matrix, a bound on the condition number is obtained from the determinant of the row equilibrated matrix as follows (Guggenheimer *et al.*, 1995):

$$\kappa_2(\tilde{\mathbf{D}}\mathbf{J}) < \frac{2}{\left|\det(\tilde{\mathbf{D}}\mathbf{J})\right|} \tag{7.73}$$

Guggenheimer *et al.* (1995) show that no tighter bound on the constant 2 is possible. From theorem 7.2, the condition number of the β-equilibrated matrix, $\kappa_2(\hat{\mathbf{D}}\mathbf{J})$, is within a factor of β of the row equilibrated matrix, so

$$\kappa_2(\hat{\mathbf{D}}\mathbf{J}) < \frac{2\beta}{\left|\det(\tilde{\mathbf{D}}\mathbf{J})\right|} \tag{7.74}$$

The determinant of the row equilibrated matrix can be evaluated simply by multiplying the pivots of the factored, row equilibrated matrix. Unfortunately, the matrix $\tilde{\mathbf{D}}\mathbf{J}$ was not formed during the scaling process; instead, the β-equilibrated matrix was formed and factored. However, the elements of $\tilde{\mathbf{D}}$ were calculated before the matrix was scaled in order to obtain $\hat{\mathbf{D}}$. Since $\tilde{\mathbf{D}} = \hat{\mathbf{D}}\check{\mathbf{D}}$, the determinant of $\tilde{\mathbf{D}}\mathbf{J}$ can be calculated as long as the elements of $\check{\mathbf{D}}$ are stored when $\hat{\mathbf{D}}$ is calculated. In fact,

$$\det(\tilde{\mathbf{D}}\mathbf{J}) = \det(\hat{\mathbf{D}}\mathbf{J})\det(\check{\mathbf{D}}) \tag{7.75}$$

The determinant of $\hat{\mathbf{D}}\mathbf{J}$ can be determined by multiplying the pivots of the β-equilibrated iteration matrix, and the $\det(\check{\mathbf{D}})$ can be determined as the row scaling factors are calculated. For the scaling that has been implemented, $1 \leq \check{d}_{ii} < \beta$.

The criterion (7.14) required that $\alpha < 1$, and we noted that if this was not satisfied, then (7.10) should be applied directly in order to estimate $||\boldsymbol{\delta x}||_{\mathrm{BDF}}$. We now argue that we can bound the value of $||\boldsymbol{\delta x}||_{\mathrm{BDF}}$ from the information provided by the evaluation and factorization of the corrector iteration matrix. During evaluation of the matrix we obtain an estimate for $\boldsymbol{\delta f}$ and during the factorization we obtain an estimate of $\boldsymbol{\delta J}$. Our estimates for these quantities will not change during the corrector iteration unless the matrix is reevaluated. Newton's method for the solution of nonlinear systems is stable when started from within the region of convergence (Wozniakowski, 1977), so we consider the rounding error introduced if we had the exact solution of the system. Numerical evaluation of the function at the exact solution $\mathbf{x}^*$ differs from zero only by the error in evaluation the function $\boldsymbol{\delta f}$:

$$\mathbf{f}(\mathbf{x}^*) = \boldsymbol{\delta f} + 0 \tag{7.76}$$

and from perturbation analysis of the system at $\mathbf{x}^*$:

$$(\mathbf{J} + \boldsymbol{\delta J})(\Delta \mathrm{x} + \boldsymbol{\delta x}) = \boldsymbol{\delta f} \tag{7.77}$$

Since we have assumed that $\mathbf{x}^*$ is the exact solution to the $\mathbf{f}(\mathbf{x}) = 0$, $\Delta\mathbf{x} = 0$. Using the perturbed system (7.77), a result analogous to (7.10) obtained:

$$\|\boldsymbol{\delta x}\|_{\mathrm{BDF}} \leq \frac{\kappa(\mathbf{J})}{\|\mathbf{J}\|_{\mathrm{BDF}} - \kappa(\mathbf{J})\,\|\boldsymbol{\delta J}\|_{\mathrm{BDF}}} \|\boldsymbol{\delta f}\|_{\mathrm{BDF}} \tag{7.78}$$

Thus, every time the corrector iteration matrix is factored and evaluated we can determine whether $\|\boldsymbol{\delta x}\|_{\mathrm{BDF}} > 1$ indicating the possibility that the desired corrector tolerance cannot be achieved even when the numerically calculated Newton update is zero. We note that we would like to converge the corrector iteration so that the numerically calculated Newton updates are less than $.33 - \|\boldsymbol{\delta x}\|_{\mathrm{BDF}}$.

7.7 Effect of Scaling

The implemented scaling technique serves two purposes. First it enables us to automatically scale models better than any user of the system could scale the models by selecting appropriate units for the system variables, because a scale factor is selected locally (in time) for each variable, rather than each type of variable (e.g., enthalpy, temperature, etc.). Second, the scaling determines the optimal condition of the system for the purposes of error analysis, and enables us to bound the condition number of the iteration matrix efficiently.

We recognize that the improvements to the performance of the code hinge upon whether the scaling affects the selection of the pivots during Gaussian elimination. Since the matrix has been scaled by integer powers of the machine base, Gaussian elimination will calculate exactly the same answer if the same pivots are selected (Forsythe and Moler, 1967). However, if the pivots change, then the answer may change as well. Therefore, the scaling helps the integration if it leads to better pivot selection during the linear algebra. The column scaling is required so that the pivot selection is attempting to minimize the backward error in the appropriate norm. The row scaling can only help the performance if it reduces the backward error of the matrix factorization. However, many linear algebra packages decide to row equilibrate matrices before attempting to factor them, so this is normally a good procedure. Since MA48 does not row equilibrate the matrix, this should help, but we cannot guarantee that it will. Since the backward error of the Gaussian elimination grows with the system size, larger systems are more likely to cause problems when the condition number is the same, and our scaling is more likely to benefit these systems.

The scaling also permits us to analyze the answers that we obtain from the Newton iteration. The accuracy of the Newton iteration is limited by the error in the residuals and the condition number of the Jacobian. The condition number that we should use in these circumstances is the minimum condition number, so we would like to have a well-scaled matrix. Our scaling provides us with a reasonably tight bound on the condition number that can be employed to detect systems in which the potential for loss of accuracy exists.

The scaling will have no effect on the performance of the integration code if the same pivots are selected, so for systems that are well scaled over the entire time domain the same performance can be expected. Since the scaling is implemented very efficiently, it will not decrease the performance. However, for poorly scaled systems, the scaling will probably change the pivots that are selected and thus change the performance of the code. This is easily seen by noting that the choice of units in which the model variables are expressed can cause simulations to fail. The ability to automatically detect the potential for inaccurate solutions due to ill-conditioning allows the code to warn users of this possibility. However, we should note that this is a worst case scenario. Note that the maximum magnification of the error in the solution is rare; both the error and the residuals must be in the appropriate directions for this to occur. Therefore, on many ill-conditioned systems, the integrator may perform quite well because the maximum amplification of the error is not observed. Our examples merely demonstrate that in some cases the amplification of the error does occur.

7.8 Conclusions

Equation-based simulation languages provide a flexible environment in which to pose dynamic simulation problems, yet this flexibility puts severe demands on the embedded numerical solution algorithms. We have found that the batch distillation of wide boiling azeotropic mixtures is a very difficult problem for the numerical integrator used within ABACUSS. In fact, we have discovered difficulties during the integration of such problems that clearly indicate that the desired solution accuracy cannot be achieved. We have proven that that these problems stem from the inability to obtain accurate solutions from the corrector iteration of the BDF integrator. Using linear error analysis, we have proven that the accuracy of the corrector depends on both the condition of the iteration matrix and the accuracy to which the residuals are evaluated. We then proved that an ill-conditioned iteration matrix can lead to the observed problems. Furthermore, we have demonstrated that even nonstiff linear time invariant ODE systems can become ill-conditioned.

We have derived a criterion under which we can ensure that the desired accuracy can be maintained and that the simulation results can be trusted. For well-conditioned systems, the BDF methods should have no problem obtaining an accurate solution as long as the residuals are accurate. However, the batch distillation examples given here do not meet this criterion, and we admit the possibility of the observed inaccurate solutions.

Since well-conditioned systems can be solved reliably, we have investigated scaling techniques to improve the conditioning of the corrector iteration matrix. Two diagonal scaling matrices, with nonzero elements that are integer powers of the machine base, are used to transform the linear system encountered at each corrector step without introducing any rounding error. We have shown that the column scaling must be chosen to reflect the error criterion imposed by the BDF integrator. Once the columns have been scaled to reflect this error criterion, we are free to choose the row scaling that minimizes the two norm condition number of the resulting system. Finding an exact minimizer of the two norm condition requires the solution of an integer programming problem. However, by extending the results of van der Sluis, we have proven that for the sparse matrices in which we are interested, we can obtain an approximate solution of this problem with a condition number that is quite close to the minimum. We have demonstrated that this approximate minimizer can be determined without even evaluating the condition number of the system. This scaling can be performed automatically and efficiently within any BDF integrator. We have implemented the algorithm within both DASOLV and DSL48S - the integration codes used within ABACUSS. The code is very efficient, making use of functions that manipulate the exponents of the binary representation of the floating point numbers, and is entirely transparent to the user of the integration code.

This numerical scaling technique has been shown to mitigate the problem of ill-conditioning on the distillation examples, reducing the condition number of the system by 14 orders of magnitude in some cases. Unfortunately, problems can always be constructed which are sufficiently ill-conditioned that the desired accuracy cannot be guaranteed with a given machine precision. In such cases, the simulation must be performed in higher precision. Note that these results apply to the dynamic simulation of any system, not just batch distillation.

Identifying potential problems in controlling the integration accuracy requires bounding the condition of the iteration matrix, the error of the evaluated residuals, and the backward error of the matrix factorization. Efficient strategies for all are required to identify and warn of potential problems automatically.

Finally, the ability to make modeling decisions that improve the condition of the DAE that is integrated has been illustrated. Future development of these ideas may focus on ways to interpret the information within the corrector iteration matrix to identify specific elements or sets of equations that may be leading to the ill-conditioning of the matrix. Proper identification of the problematic terms may permit the user to reformulate the model, if suitable modeling assumptions can be made without sacrificing the applicability of the results, in a way that enables the numerical routines to perform better. In addition, symbolic techniques capable of reducing the error in the calculated residuals may also increase the number of problems that can be solved reliably, and these should be investigated further.

Chapter 8

Initial Step Size Selection for Differential-Algebraic Systems

8.1 Introduction

The transient behavior of many physical systems of interest exhibits both continuous and discrete characteristics. On the one hand, continuous behavior is naturally formulated mathematically as differential-algebraic equations (DAEs) (Pantelides *et al.*, 1988; Brenan *et al.*, 1996; Mattsson, 1989; Cellier and Elmqvist, 1993), and on the other, discrete behavior is typically the result of either external control actions or autonomous discontinuities (Barton and Park, 1997). Mathematically, discrete aspects of the system behavior are modeled as changes in the functional form of the underlying DAE. The existence of such discontinuities complicates the solution procedure and increases the need to start integration codes efficiently.

The solution of an initial value problem described by DAEs containing discontinuities can be formulated as a combined discrete/continuous simulation problem (Cellier, 1979; Barton and Pantelides, 1994). In fact, the mathematical formulation of this problem is typically represented as a sequence of initial value problems containing continuous models. Discontinuities, commonly known as events, define the boundaries between these continuous domains and may result in a discrete change to either the variable values, the functional form of the model, or both. Thus, the simulation domain of interest $[t_o, t_f)$ is partitioned into NC continuous sub-domains $[t^{(k-1)}, t^{(k)})\ \ \forall\, k = 1 \ldots NC$ in which $t_o = t^{(0)}$ and $t_f = t^{(NC)}$. The combined simulation problem is defined as follows:

$$\left.\begin{aligned} f^{(k)}(x^{(k)}, \dot{x}^{(k)}, y^{(k)}, u^{(k)}, t) = 0 \\ u^{(k)} = u^{(k)}(t) \end{aligned}\right\} \quad t \in [t^{(k-1)}, t^{(k)})\ \ \forall\, k = 1 \ldots NC \tag{8.1}$$

where $x^{(k)} \in \mathbb{R}^{n_x^{(k)}}$, $y^{(k)} \in \mathbb{R}^{n_y^{(k)}}$, $u^{(k)} \in \mathbb{R}^{n_u^{(k)}}$, and $f^{(k)} : \mathbb{R}^{n_x^{(k)}} \times \mathbb{R}^{n_x^{(k)}} \times \mathbb{R}^{n_y^{(k)}} \times \mathbb{R}^{n_u^{(k)}} \times \mathbb{R} \rightarrow \mathbb{R}^{n_x^{(k)} + n_y^{(k)}}$. The event times $t^{(k)}$ may be defined either explicitly (time events) or implicitly (state events) during the course of the simulation. If all of the discontinuities are defined explicitly, the solution of each of the initial value problems may proceed in a straightforward fashion (Ellison, 1981). Otherwise, the time at which these events occur must be determined simultaneously with the solution of the initial value problems; an efficient algorithm for detecting and locating state events within a linear multistep method has been developed by Park and Barton (1996).

In any case, the solution of a single combined simulation problem may require the solution of many initial value problems. Integration codes with the ability to handle stiff systems, such as BDF methods, automatically adjust both the step size and the order to produce an accurate solution efficiently. To maintain credibility of the error estimates, which are based on the local error in the solution, the step size control permits only moderate changes in the step length on any given step. Practical experience has shown that this strategy permits an efficient solution once the step size is 'on scale' for the problem. This implies that the step size chosen at the beginning of each sub-domain should be 'on scale' for the current system dynamics. If the initial step is not chosen properly, the step size control is quite inefficient at finding a value that is on scale. Moreover, the error estimates may fail to recognize an unacceptable solution when the step size is not on scale. Since the integrator will be started many times during a combined simulation experiment, the reliability and efficiency of the initial phase of the integration algorithm can have a significant impact on the performance of the overall solution procedure.

This chapter derives an efficient method to start the integration code for an arbitrary initial value problem in DAEs, corresponding to any particular instance (k) in (8.1). Since this work has been motivated by combined simulation problems, the method has been tailored for the calculation sequence encountered during the combined simulation of DAE models (Park and Barton, 1996) and the information that is readily available in combined simulation environments such as ABACUSS[1] and gPROMS (Barton, 1992). The method applies to DAE systems with index ≤ 1 for which a consistent initial condition is known.

Next, we examine the heuristics commonly used to select the initial step length within ODE and DAE codes, and examine methods that have been employed to improve upon these heuristics for ODE codes. We then consider how some of the fundamental differences between DAEs and ODEs (Petzold, 1982b) affect the initial phase of the integration code. For example, the information available at the start of the integration and the form of the equations to be solved preclude the direct extension of the ODE methods. However, since the underlying problem is similar, the same basic ideas used to increase efficiency and reliability at the start of the integration apply. In particular, the method we propose addresses the differences that exist in the specification of initial conditions for DAE systems. We exploit the facts that a consistent initialization calculation must be performed before the integration method is called, and that expressions for the partial derivatives of the equations are now commonly available within combined simulation environments.

8.2 Initial Step Size Selection

The initial step size that is selected must be 'on scale' for the problem under consideration. It should be small enough to capture the dynamics of interest within the requested accuracy, yet it should not be so small that it significantly affects the efficiency of the solution. A number of authors have addressed the selection of initial

[1] ABACUSS (Advanced Batch and Continuous Unsteady-State Simulator) Process Modeling Software, a derivative work of gPROMS Software, Copyright 1992 by the Imperial College of Science, Technology and Medicine.

step length in codes used to solve ordinary differential equations (Gear, 1980a; Watts, 1983; Gladwell *et al.*, 1987; Shampine, 1987), and a more complete description of the previous work can be found there. Here we address the heuristics for initial step size selection contained within popular DAE codes.

Several rules of thumb are commonly employed to select the initial step size in codes designed to solve ordinary differential equations. The simplest strategy is to require the user to provide an initial step size. Another technique seen in practice is to calculate the length of the initial step as a (fixed) fraction of the length of the first output interval. These are two of the strategies implemented within DASSL (Petzold, 1982a); if the user does not supply a value, DASSL defaults to either a fraction of initial output length or the inverse of the norm of the variable derivatives (Shampine and Gordon, 1972), whichever is smaller. Allowing the user to specify the initial step size permits educated users to exploit knowledge about the specific system they want to solve, but most users will supply a somewhat arbitrary value because they may not have a good idea of what an appropriate initial step length is. On the other hand, the length of the first output interval should provide some indication of the scale of the problem. However, as Watts (1983) discusses, the user may not care about the initial behavior of the solution, so the first output interval may not reflect the initial dynamics of the problem. Furthermore, this criterion does not even consider the solution accuracy desired. Using the norm of the variable derivatives is more sensible; however, the time derivatives of the algebraic variables are not required to specify a consistent set of initial conditions for the DAE, and for systems evolving from a steady state the time derivatives are initially zero.

To ensure accuracy during the initial step, Sedgwick (1973) suggested starting the integration at the smallest permissible step size given the machine precision. The integrator will then steadily increase the step size until it reaches a reasonable value. Since most codes do not permit the step size to change too rapidly, with this approach many steps will probably be required before the step size levels off at a reasonable value. For example, DASSL only permits the step size to increase by a factor of two on any step where an increase in step size is desired, in order to insure that the error estimates remain valid (Brenan *et al.*, 1996). For linear multistep methods, a doubling of the step size often requires refactorization of the corrector iteration matrix. Therefore, starting with too small a step size will incur unnecessary computational costs (for a dramatic illustration of this, see the bouncing ball example in section 8.9). In addition, the asymptotic error estimates may become so contaminated with roundoff errors that they prevent the step size from increasing as it should (Watts, 1983), reducing the efficiency of the integrator even further. This phenomenon is likely to be magnified when dealing with DAE systems, since the condition number of the iteration matrix scales as $(1/h)$ for index-1 DAEs (Petzold, 1982b), implying that the accuracy of the solution to the linear system solved at each corrector iteration is more sensitive to rounding errors when the step size is small.

If an initial step that is too large is attempted, the user relies on the integrator to reduce the step size until the error criterion is satisfied. Such a situation arises when the initial step length is selected as a fraction of the initial output interval and the user is not particularly interested in the initial behavior of the solution. Several problems may result from such an approach. The asymptotic error estimates may not be valid for the large step sizes attempted initially. In such cases, the predictor will not be close to the true solution and may cause the corrector iteration to fail. The step size is reduced and the procedure is repeated until the corrector converges and the integration tolerance is satisfied. For linear multistep methods, the heuristics typically require refactorization of the corrector iteration matrix after a failed corrector iteration or a significant step reduction, so successive step reductions are inefficient. In addition, the possibility exists that the error criterion could be satisfied at a step size which is too large for the asymptotic error estimates to be valid. In such a case, some local phenomena may be missed entirely (Watts, 1983). For example, the norm of the difference between the predicted and corrected solutions may not be a unimodal function; this implies that a solution that satisfies the error tolerances may exist for which the corrector polynomial does not accurately represent the true solution over the initial interval. Although this phenomenon is probably rare, the initial step size selection procedure should avoid such situations.

More sophisticated strategies to select the initial step size have been developed. These strategies are based on estimates of the norm of the variables' derivatives (Shampine and Gordon, 1972), the value of the local Lipschitz constants (Shampine, 1980) for the system, or the norm of the higher order derivatives (Watts, 1983; Gladwell *et al.*, 1987; Shampine, 1987) of the variables at the initial time. These methods are concerned with both stability and accuracy when selecting the initial step size, but in most cases it is assumed that the equations will not be stiff at the initial conditions. Although these ideas are applicable to linear multistep methods, most of this work has focused on the application of one-step methods to explicit systems of ODEs. An estimate of the behavior of the solution at the initial time is developed, and this estimate is used to find an appropriate initial step size. In most cases, these methods rely on the existence of explicit expressions for $\dot{x}$ in terms of x and t.

In this work, we follow the basic idea of deriving an approximation for the behavior of the solution at the initial condition, but the treatment of fully implicit DAE systems (8.1) requires a different approach to derive estimates of the initial solution behavior. In the next section, we highlight the differences between the explicit ODE systems addressed in the past and the DAEs with which we are concerned.

8.3 Scope

This work addresses the initial phase of the integration of index-1 DAE systems in implicit form using a linear multistep method. The systems considered are those defined in (8.1) corresponding to a particular instance (k). We determine an efficient step size to be used during the first integration step on which a first order linear multistep method is employed. Consistent initial values (see section 8.5) $\dot{x}(t_o)$, $x(t_o)$, and $y(t_o)$ are supplied to the integration routine. These values are the result of a consistent initialization calculation performed before the integrator is called. Kröner *et al.* (1992) have shown that failure to provide consistent initial conditions will result in a myriad of problems including possible failure of the integrator on the first step and inaccurate solution of the problem. In addition, routines to evaluate the partial derivatives of f, and the derivatives of the input functions, du/dt, are supplied.

We do not consider any of these requirements as serious limitations of our approach because we envision the primary application for this technique to be integration codes embedded within modern combined simulation environments. Within such environments the functional form of the model is explicitly available, so the partial derivatives can be calculated automatically and efficiently (Tolsma and Barton, 1997). Since the user can specify a DAE model of arbitrary index within these systems, we advocate the calculation of the consistent initial condition as a separate phase of the solution procedure; the structure of the model is analyzed in this phase of the calculation. First, the equations that define a consistent initial condition are identified and solved (Pantelides, 1988; Feehery and Barton, 1996a). Next, if the system is high index, in most practical cases an equivalent index-1 DAE can be derived automatically (Feehery and Barton, 1996a). Hence, even in the high index case, it can be assumed that an index-1 system will always be passed to the numerical integration code for solution. Previous researchers have determined conditions under which the solutions of the reinitialization problems required at the junctions of the simulation domains ($t = t^{(k)}$) are defined unambiguously (Brüll and Pallaske, 1991; Brüll and Pallaske, 1992; Barton and Park, 1997).

BDF (Gear, 1971) integration codes have been shown to be efficient and highly reliable for the solution of index-1 DAEs, so these are typically employed within simulation environments. It is possible to start these methods at a higher order by using one-step methods, such as a fourth order Runge-Kutta method (Gear, 1980a; Gear, 1980b; Brankin *et al.*, 1988). However, these techniques are most applicable when the system is not stiff and an explicit RK method can be employed. The applicability of implicit RK methods for the same purpose is questionable because a set of p nonlinear systems of equations must be solved to start a pth order method (Gear, 1980b). In many situations, the systems are not stiff during the initial portion of the integration because the fast transients in the system are excited and the step size is chosen based on accuracy rather than stability requirements (Lambert, 1991), but this property is not guaranteed. In a simulation environment we wish to emphasize the reliability of the numerical solution, and to minimize the need for user intervention in tuning the solution process. Hence, to ensure stability, we employ a first order BDF method which is A-stable (Hairer and Wanner, 1993); this also permits us to take advantage of the order selection strategies within DASSL.

Hybrid techniques employing an explicit RK method initially that switches to a BDF scheme for stability (Keeping, 1995) were not considered in order to retain the guarantees for the detection of state events provided by the method of Park and Barton (1996). In cases where state events are guaranteed not to occur in the initial phase of the integration these methods may be effective, but guarantees concerning stability and state event location cannot be provided in general.

8.4 Methodology

A higher order approximation of the behavior of the solution at the initial time is employed to start the first order BDF method efficiently. This approximation estimates the difference between the first order method and the true solution to provide an estimate of the initial step size. The estimate is then employed to advance the solution over the initial integration step and to solve for the length of this step simultaneously. The method consists of the following steps:

1. Determine the derivatives of the algebraic variables $\dot{y}_o$ at the initial time. The second derivatives of the differential variables $\ddot{x}_o$ are also obtained.

2. Estimate the value for the initial step size.

3. Advance the simulation over the first integration step, calculating the initial step size and the variable values simultaneously.

The objectives of the final two steps in this procedure are similar to those of Gladwell *et al.* (1987) and Shampine (1987); these employ the basic concept proposed by Sedgwick (1973) in a more efficient fashion. The procedure is implemented by modifying the integrator's behavior over the first integration step. Our approach differs from those employed for ODEs because we are dealing with fully implicit DAE models, and we assume that the Jacobian will be available.

The first step is required because the consistent initial condition for the index-1 DAE passed to the integration code does not define $\dot{y}$; on the other hand, for the explicit ODE case, the time derivatives of all the variables are always available from a function evaluation. The benefits obtained by using this information for the first order prediction are demonstrated in section 8.9. The estimate of $\ddot{x}$ also derived at this step provides a convenient way to estimate the initial step size (h_{est}) at the second stage of our procedure. This contrasts with the initial estimates employed for ODE codes in which only first derivative information is typically available at the start. Attempts to obtain more information by taking small steps are complicated by the fact that while the truncation error is reduced as the step size decreases, the relative contribution of rounding error is increased as the step size decreases.

The last step in the procedure involves the solution of a nonlinear system of equations to determine the optimal initial step size and the solution of the DAE at this time simultaneously. The availability of the Jacobian matrix and an estimate for the optimal step size from the previous steps in our method enables this system of equations to be solved using a modified Newton iteration. The solution of this system of equations satisfies the DAE model and the criteria employed to define the optimal initial step size, which are described in section 8.7. Note that these criteria consider the step size and order selection heuristics employed by the integration code. While the examples provided within this paper employ the heuristics of the integration code DASSL (Brenan *et al.*, 1996), the same ideas apply to other linear multistep methods. At the conclusion of this step, we verify that the estimate of the local truncation error decreases as the step size is reduced to support the assumption that we have determined the first point at which the desired error norm is attained.

The method outlined above exploits the fact that a consistent initialization calculation has been performed in order to derive a linear system to calculate the algebraic derivatives. Although solution of this linear system is not required (a zero order approximation for the algebraic variables could be employed in the predictor), availability of the derivatives of the algebraic variables enables much larger initial step sizes to pass the truncation error tolerance. In cases where any of the algebraic variables are changing significantly at the initial time, the zero order approximation will not be very accurate. This will result in a large difference between the predicted and corrected solution on the initial integration step, requiring a very small initial step in order to meet the error criterion. Many additional steps are then required to increase the step size to the value that might have been possible if these derivatives were available. In contrast, the algebraic derivatives can be determined with little computational effort. The benefits that this calculation has on the efficiency of the initial stages of the integration is demonstrated on a collection of example problems detailed in section 8.9.

The remainder of this chapter discusses each of the steps described above in more detail. First, the consistent initialization calculation is reviewed since the derivation of the system employed in step 1 of our procedure relies upon the equations used to determine the consistent initial condition. This method is then compared with the heuristics currently employed within DASSL, demonstrating the benefits of employing this technique within combined simulation environments.

8.5 Consistent initial conditions

Before the integration of a system of DAEs can begin, a set of consistent initial conditions must be defined. These are represented by initial values for the differential variables x, their derivatives $\dot{x}$, and the algebraic variables y that satisfy the model equations, their first and higher order time derivatives, and an additional set of specifications enforced at the initial time. The additional specifications take up the degrees of freedom that remain when all constraints on $\dot{x}$, x, and y implied by the DAE model and its time derivatives are taken into account. The ability to express the initial conditions in terms of general algebraic relationships between the model variables at the initial time, rather than simply specifying initial values for a subset of the variables, is required to formulate many simulation problems of interest (Barton and Pantelides, 1994). This work considers index-1 DAEs (8.3) for which the following matrix has full rank:

$$\left[\frac{\partial f}{\partial \dot{x}}\ \frac{\partial f}{\partial y}\right] \tag{8.2}$$

When the matrix in (8.2) has full rank, the model equations (8.3) and additional initial specifications (8.4) need to be solved simultaneously to determine initial values $\dot{x}_o = \dot{x}(t_o)$, $x_o = x(t_o)$, and $y_o = y(t_o)$:

$$\begin{aligned} f(\dot{x}_o, x_o, y_o, u(t_o), t_o) &= 0 && (8.3) \\ c(\dot{x}_o, x_o, y_o, u(t_o), t_o) &= 0 && (8.4) \end{aligned}$$

Therefore, $c : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_y} \times \mathbb{R}^{n_u} \times \mathbb{R} \to \mathbb{R}^{n_x}$. We refer to the solution of (8.3–8.4) as a set of consistent initial conditions. Note that full rank of the matrix shown in (8.2) is a sufficient condition for the index ≤ 1.

The index one DAEs considered in this work represent models that are either naturally expressed as index-1 differential algebraic systems, or that are a member of the family of equivalent index-1 systems corresponding to a model that is naturally expressed as a high index (i.e., index ≥ 2) DAE. The equivalent index-1 models considered have been derived from the application of the dummy derivative algorithm (Mattsson and Söderlind, 1993), which can be applied to high index models automatically (Feehery and Barton, 1996a). This algorithm yields a DAE whose structural index is one; for all such systems, the matrix appearing in (8.2) is structurally nonsingular (Duff *et al.*, 1986). The algorithm can also be applied to the class of special index-1 DAEs (Pantelides, 1988) that do not satisfy (8.2) based on structural criteria. Thus, the implementation described here applies to all index-1 systems for which structural criteria can correctly determine the additional equations constraining the initial condition.

Consistent initial conditions are obtained by solving $f(\dot{x}_o, x_o, y_o, u_o, t_o) = 0$ and $g(\dot{x}_o, x_o, y_o, u(t_o), t_o) = 0$. Typically an initial guess that is close to the solution is provided, either from a physical analysis or a solution obtained using another numerical strategy such as homotopy continuation, and a modified Newton method is used to converge the system; we assume that an appropriate guess is provided, so the method will succeed. Implementation of this method requires the partial derivatives of the f and g with respect to $\dot{x}$, x, and y. These derivatives are easily calculated by applying symbolic (Hearn, 1987) or automatic differentiation techniques (Hillstrom, 1985; Bischof *et al.*, 1992) to the functions f and g, so they are readily available within equation based modeling environments such as SpeedUp, ABACUSS, and gPROMS (AspenTech, 1993; Barton, 1992).

The convergence criterion specified for the Newton iteration must take into account the way in which error in the solution to the DAEs will be measured by the integrator. At the very least, the size of the final Newton updates for x and y must satisfy the truncation error criterion employed on the first integration step to enable the first integration step to proceed. To ensure that the desired accuracy can be achieved during the integration, the distance of the numerical approximation from the exact solution to the initialization problem should be controlled; the impact that the termination criterion has on accuracy of the numerical solution is discussed in chapter 7 and elsewhere (Allgor and Barton, 1997a; Iri, 1988). Insufficiently accurate convergence of the initialization problem will lead to the same type of numerical difficulties caused by an inconsistent initial condition (Kröner *et al.*, 1992).

It is therefore assumed that the values of $\dot{x}_o$, x_o, and y_o at the conclusion of the Newton iteration provide sufficiently accurate consistent initial conditions for the solution of the DAE model (8.3). Theoretically, no more information is required to start the integration. However, the integrator can be started more efficiently if $\dot{y}_o$ is provided. The next section demonstrates that both $\dot{y}_o$ and $\ddot{x}_o$ can be calculated quite cheaply using a portion of the Jacobian matrix employed during the initialization calculation.

8.6 Derivatives of algebraic variables

The derivative of the DAEs with respect to time determines $\dot{y}_o$ and $\ddot{x}_o$. We fix $\dot{x}_o$, x_o, and y_o at the values calculated during initialization, and solve the following linear system for the values of $\ddot{x}$ and $\dot{y}$:

$$\frac{\partial f}{\partial \dot{x}}\frac{d\dot{x}}{dt} + \frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial y}\frac{dy}{dt} + \frac{\partial f}{\partial u}\frac{du}{dt} + \frac{\partial f}{\partial t} = 0 \tag{8.5}$$

Noting that $\ddot{x} = d\dot{x}/dt$ and $\dot{y} = dy/dt$ and rearranging (8.5) produces the following system of equations:

$$\frac{\partial f}{\partial \dot{x}}\ddot{x} + \frac{\partial f}{\partial y}\dot{y} = -\frac{\partial f}{\partial x}\dot{x} - \frac{\partial f}{\partial u}\frac{du}{dt} - \frac{\partial f}{\partial t} \tag{8.6}$$

which can be evaluated at the initial time to produce the following linear system whose solution defines the new variables:

$$\begin{bmatrix} \frac{\partial f}{\partial \dot{x}} & \frac{\partial f}{\partial y} \end{bmatrix}_{t=t_o} \begin{bmatrix} \ddot{x}_o \\ \dot{y}_o \end{bmatrix} = -\left.\frac{\partial f}{\partial x}\right|_{t=t_o} \dot{x}_o - \left.\frac{\partial f}{\partial u}\right|_{t=t_o} \dot{u}(t_o) - \left.\frac{\partial f}{\partial t}\right|_{t=t_o} \tag{8.7}$$

Note that all of the partial derivatives appearing in (8.7) are defined entirely in terms of quantities that have already been calculated (i.e., $\dot{x}_o$, x_o, y_o, and t_o) or are known (i.e., $\dot{u}(t_o)$). In addition, $\partial f/\partial \dot{x}$, $\partial f/\partial y$, and $\partial f/\partial x$ were evaluated during the Newton iteration employed to determine the initial conditions, and we have assumed that a routine that returns them is available; alternatively, these quantities could be calculated using finite differences since u is an explicit function of t and f is an explicit function of $\dot{x}$, x, y, and t. These matrices are simply evaluated using $\dot{x}_o$, x_o, y_o, and t_o. The remaining terms on the right hand side require the derivatives of the input functions appearing in the DAE; these can be derived and evaluated using automatic differentiation techniques (see appendix C for the derivation of the linear system defining the derivatives of the algebraic sensitivity variables.). In typical applications, (8.7) defines a sparse unstructured linear system that can be solved efficiently (Duff and Reid, 1993; Duff and Reid, 1995; Harwell, 1993).

To guarantee that (8.7) can be solved to determine unique $\ddot{x}_o$ and $\dot{y}_o$, the matrix on the left hand side must be nonsingular. During the consistent initialization, we can check that the matrix shown in (8.7) is structurally nonsingular (Duff *et al.*, 1986) (deriving an equivalent index-1 system for which this holds by applying the method of dummy derivatives, if necessary). Thus, for any DAE system to which we have obtained a consistent set of initial conditions for the equivalent index-1 model, we will have a structurally nonsingular matrix in (8.7). However, this matrix may still be singular, so we need to check the pivots of this factored matrix as we attempt to solve this system. Singularity of this matrix is not sufficient to show that (8.3) is even locally index ≥ 2, but it raises the suspicion that the index of the system and/or the degrees of freedom for consistent initialization cannot be properly determined using structural criteria. In these situations, the code terminates with a warning indicating the strong suspicion that the model is still high index despite any attempts at index reduction. For example, consider the following linear constant coefficient DAE:

$$\dot{x}_1 - x_1 + x_2 - y_1 = 0 \tag{8.8}$$

$$\dot{x}_2 - x_1 - x_2 + y_1 = 0 \tag{8.9}$$

$$x_1 + x_2 = 0 \tag{8.10}$$

The combination of Pantelides' algorithm and the method of dummy derivatives will yield the following system:

$$\dot{x}_1 - x_1 + y_2 - y_1 = 0 \tag{8.11}$$

$$y_3 - x_1 - y_2 + y_1 = 0 \tag{8.12}$$

$$x_1 + y_2 = 0 \tag{8.13}$$

$$\dot{x}_1 + y_3 = 0 \tag{8.14}$$

where x_2 and $\dot{x}_2$ have been replaced by the algebraic variables y_2 and y_3. The matrix

$[\partial f/\partial \dot{x}\ \partial f/\partial y]$ is given by:

$$\begin{bmatrix} 1 & -1 & 1 & 0 \\ 0 & 1 & -1 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

which, while structurally nonsingular, is still singular. In the linear time invariant case, this indicates that the structural algorithm has underestimated the true index of this DAE, which is 3. Unfortunately, in general, no conclusions can be drawn about the index of general nonlinear DAE systems based on the singularity of this matrix, but we can suspect that the index has been underestimated.

Factorizing the matrix on the left hand side of (8.7) dominates the computational cost of determining $\ddot{x}_o$ and $\dot{y}_o$. Since this matrix is smaller than the Jacobian matrix used in the Newton iteration during initialization, the additional cost of calculating $\dot{y}_o$ and $\ddot{x}_o$ is expected to be small compared to the effort required to solve the initialization problem.

8.7 Initial step size

The integration will start using a first order method, so the initial step length can be determined based on accuracy requirements alone, since the first order BDF method is stable. We consider the accuracy criteria when choosing the initial step size. In particular, larger steps will lead to a more efficient solution if the accuracy can be maintained using larger step sizes.

The heuristics used to control the step size within DASSL adjust the step size based on the estimate of the local error. These error estimates are asymptotically correct in the case of constant step size and order (Brenan *et al.*, 1996), so the heuristics within DASSL favor sequences of steps at constant size and order. We employ the following criteria to identify a step size to use on the initial step that will lead to an efficient integration:

1. The initial step must satisfy the requested error tolerances.
2. The length of the second integration step must be the same size or greater than the length of the initial step.
3. The norm used to measure local error in the solution must accurately represent the deviation from the predicted solution, i.e., the first order approximation should interpolate the solution to within the requested tolerances over the domain of the initial step.

These criteria warrant some explanation. The requested error tolerances are enforced using the weighted norm of the difference between the predicted and corrected solution. Based solely on this criterion, the maximum step that satisfies the local error criterion would be selected.

The second criterion limits the size of the initial step in order to ensure that the next step can be carried out at the same size. The purpose of this criterion is to take advantage of the heuristics employed within multistep methods that favor sequences of steps of constant length and order. The initial step size considers the heuristic used to determine the length of the subsequent step. For example, DASSL employs a conservative strategy to select the size of the next step in order to limit the number of truncation error failures; therefore, a fairly aggressive initial step can pass the convergence tolerance, but the size of the succeeding step will be reduced. In addition, successive steps of the same size are typically required to increase the order of the integration method. DASSL only considers increasing the order of approximation of a kth order method after $k+1$ successful steps of the same length at order k. We assume that the potential benefits afforded by increasing the integration order outweigh any advantage that may be obtained by taking a slightly larger initial step. Since the second step will also employ a first order approximation, we will see that the conservative step size heuristics dictate that our second criterion is more restrictive than our first.

The third criterion is included to ensure that no local phenomena in which we are interested are missed because the initial time discretization is too coarse. The first order approximation to the solution is asymptotically correct, but the initial step size needs to be small enough so that the local estimate of the error represents the divergence from this asymptotic limit. Therefore, the first point in time at which the local error reaches the value defined by the second criterion is desired. Although we cannot guarantee that some important phenomena have not been missed, we select the initial step size in a way that attempts to ensure that the first order approximation properly interpolates the solution over the initial step. Over this region, we expect the norm of the difference between the predicted and corrected solution to increase with step size, and we can easily evaluate the derivative of the error norm with respect to step size at the completion of the initial step. However, ensuring that important phenomena are not overlooked is difficult, because the numerical accuracy of the solution to the model equations tends to decrease as the step size is decreased because the corrector matrix becomes more ill-conditioned as h approaches zero for index-1 DAEs.

We refer to a step of maximum length that meets these criteria as the *optimal initial step size* h_{opt} in the remainder of this chapter; note that our definition differs slightly from the definition of h_{opt} used by Gladwell (1979) and Watts (1983) due to the introduction of the second and third criteria. Equations that define h_{opt} according to the first two criteria are derived in the following section. We demonstrate that these equations can be solved during the first integration step by augmenting the system of equations solved during the corrector iteration.

8.7.1 Defining the optimal initial step size

Although the consistent initialization calculations distinguish between the differential and algebraic variables of the model, the integration code makes no such distinction. For convenience, the model equations are defined in terms of a single vector of variables throughout the remainder of the paper. Let $n_z = n_x + n_y$, $z^T = [x^T y^T]$, and $\dot{z}^T = [\dot{x}^T \dot{y}^T]$. Rather than defining a new function, we let the function f operate on the vectors z and $\dot{z}$ with the assumption that $f(\dot{x}, x, y, u, t) = f(\dot{z}, z, u, t)$ where by the definition of a DAE, $\partial f / \partial \dot{z}$ is singular everywhere. The first criterion defining the optimal initial step size is satisfied by any solution to the following system of nonlinear equations for $\epsilon \in \mathbb{R}$ such that $0 \leq \epsilon < 1$:

$$\begin{aligned} f((z^C - z_o)/h_o, z^C, u(t_o + h_o), t_o + h_o) &= 0 \qquad (8.15) \\ f_\tau(z^C, h_o, \epsilon) = M \left\| z^C - z^P \right\|_{\text{BDF}} - 1 + \epsilon &= 0 \qquad (8.16) \end{aligned}$$

where M is a constant associated with the integration method and z_o is the solution of the DAE at initial time t_o. For example, M has a value of 1/2 at the conclusion of the first integration step for the fixed leading coefficient BDF method. The parameter ϵ represents the approach to the limit of acceptable error. The norm used to evaluate the error in the solution $\|\cdot\|_{\text{BDF}}$ is the weighted root mean square norm defined by (6.7) and repeated below for convenience:

$$\|z\|_{\text{BDF}} = \sqrt{\frac{1}{n_z} \sum_{i=1}^{n_z} \left| \frac{z_i}{\tau_{ri} \left|\tilde{z}_i\right| + \tau_{ai}} \right|^2}$$

where the vector $\tilde{z}$ takes the values of z from the previous time step.

The value of h at the solution of (8.15–8.16) with $\epsilon = 0$ corresponds to the definition of h_{opt} defined by Gladwell (1979) and Watts (1983). In our case, the requirement that the second step will not be smaller than the first defines the value of ϵ. The second step taken by DASSL will be another first order step, so the heuristic used to suggest the size for for the next step reduces to the following (Brenan *et al.*, 1996):

$$h_1 = \frac{h_o}{2M \left\| z^C - z^P \right\|_{\text{BDF}}} \qquad (8.17)$$

When h_1 is chosen according to this heuristic, the second criterion defining the optimal initial step size ($h_1 \geq h_o$) is satisfied as long as $\epsilon \geq 1/2$. Thus, $\epsilon = 1/2$ provides the maximum length initial step that satisfies the first two criteria.

The predicted values of the solution are defined by the first order approximation $z^P = z_o + h_o \dot{z}_o$, using the values of z_o and $\dot{z}_o$ calculated during the solution of (8.3), (8.4), and (8.7). Equations (8.15–8.16) are then solved for z^C and h_o using a modified Newton method that permits the use of a deferred Jacobian. Initial guesses for z^C and h_o must be provided in order for the method to converge. A method to estimate an initial guess for h is discussed in the next section. This estimate seeks to find the smallest value of h for which (8.15–8.16) hold in order to satisfy the third criterion above.

8.7.2 Initial step size estimator

Any solution of (8.15–8.16) satisfies the first two criteria for h_{opt} provided ϵ corresponds to the step size heuristics of the particular code. We also desire a value h_o for which the error estimate is also valid, noting that the values of h_o satisfying (8.15–8.16) may not be unique.

For small h_o, the difference between the solution predicted by the linear approximation at t_o and the exact solution of the DAE is given by the higher order terms in a Taylor series expansion about the initial point:

$$z(t_o + h_o) - (z_o + h\dot{z}_o) = z(t_o + h_o) - z^P = \frac{h^2}{2}\ddot{z}_o + O(h^3) \tag{8.18}$$

The BDF method approximates the exact solution $z(t_o+h_o)$ with the solution of (8.15) z^C. The integration code maintains the validity of the approximation by controlling the local truncation error, so the calculated solution z^C obeys a similar relationship:

$$z^C - (z_o + h\dot{z}_o) = z^C - z^P \approx \frac{h^2}{2}\ddot{z}_o + O(h^3) \tag{8.19}$$

From (8.19), we estimate the quantity $z^C - z^P$ used to define the local truncation error in (8.16) using $h^2\ddot{z}_o/2$. Using this approximation, an estimate of the initial step size that satisfies (8.16) is given as follows:

$$h_{\text{est}} = \sqrt{\frac{2\,(1-\epsilon)}{M\,\|\ddot{z}_o\|_{\text{BDF}}}} \tag{8.20}$$

The error estimate is credible if the second term in the Taylor series dominates the higher order terms in the series over a step of length h_{est}. The algebraic derivative calculation provides $\ddot{x}_o$, but $\ddot{y}_o$ is required to define $\ddot{z}_o$ completely. Since only the norm of $\ddot{z}_o$ is required to estimate the initial step size and $\|\cdot\|_{\text{BDF}}$ scales with the square root of the number of elements in the vector, we assume that $\|\ddot{z}_o\|_{\text{BDF}} = \|\ddot{x}_o\|_{\text{BDF}}$ as an initial approximation. At the conclusion of the initial integration step, we attempt to verify that the difference between the predicted value and the solution z^C at h_{est} for the differential variables is approximated by the second order term in the Taylor series indicating that the contributions from the higher order terms are negligible.

8.7.3 Initial time step combined with step size selection

The variable values and the optimal size for the initial step are simultaneously determined during the first integration step. The nonlinear system (8.15–8.16) is solved for $[z(h_o), h_o]^T$ using a modified Newton iteration in which a deferred Jacobian is employed.

The linear system solved at each step of the standard corrector iteration (i.e., if h_o were specified) on the initial integration step follows:

$$\left[\frac{\partial f}{\partial z} + \frac{1}{h_o}\frac{\partial f}{\partial \dot{z}}\right]\left[\Delta z^k\right] = G\Delta z^k = -f((z^k - z_o)/h_o, z^k, u, h_o) \tag{8.21}$$

In order to solve for both $z(h_o)$ and h_o simultaneously, we solve the following system at each step of the Newton iteration:

$$\begin{bmatrix} \frac{\partial f}{\partial z} + \frac{1}{h_o}\frac{\partial f}{\partial \dot{z}} & -\frac{z^k - z_o}{h^2}\frac{\partial f}{\partial \dot{z}} + \frac{\partial f}{\partial u}\frac{\partial u}{\partial t} + \frac{\partial f}{\partial t} \\ \frac{\partial f_\tau}{\partial z} & \frac{\partial f_\tau}{\partial h} \end{bmatrix} \begin{bmatrix} \Delta z^k \\ \Delta h^k \end{bmatrix} = J \begin{bmatrix} \Delta x^k \\ \Delta h^k \end{bmatrix} = \begin{bmatrix} -f((z^k - z_o)/h, z^k, u, h^k) \\ -f_\tau(z^k, h^k, \epsilon) \end{bmatrix} \tag{8.22}$$

Observe that the standard corrector iteration matrix G is equivalent to the first n_z rows and columns of the Jacobian matrix J used for the modified corrector iteration. On large problems, factoring the corrector iteration matrix dominates the computational cost of the integration method, so BDF codes avoid factoring the corrector iteration matrix at each time step by employing the already factored corrector iteration matrix from a previous time step until the convergence of the Newton iteration deteriorates to an unacceptable level. This implies that as long as the guess for h_{est} is reasonably close, the same Jacobian matrix can be used throughout the entire Newton iteration, and that only the matrix G should be factored, so it can be used on the subsequent integration step without requiring a refactorization.

For the type of systems in which we are interested, the matrix G is sparse and unstructured, so the integration code employs linear algebra routines that take advantage of this (Duff and Reid, 1993; Duff and Reid, 1995). Although the additional row contained in J is dense, the matrix J remains sparse. The structure of J is exploited by factoring only the matrix G and by treating the last row and column of J separately. At each Newton iteration (8.22) can be solved for the cost of two backsubstitutions on a system of size n_z and a couple of dot products; the solution procedure is described in appendix B. The main reason to avoid forming and factoring J is to avoid having to refactor the corrector matrix on the next integration step. However, some additional benefits are obtained by exploiting the fact that the additional row contained in J is dense. The dense row in J removes any block diagonal structure from J that may have existed in G. Treating the additional row separately takes full advantage the block structure of G, which is particularly important for the simultaneous integration of a DAE system and its parametric sensitivities; for these systems, efficient solution techniques have been developed that exploit the fact that the linear systems encountered will block decompose (Maly and Petzold, 1996; Feehery *et al.*, 1997).

The derivative expressions appearing in the last column of J were required to compute the derivatives of the algebraic variables in section 8.6, so routines to provide them are assumed. Let D define the diagonal matrix of variable weights for the root mean square norm:

$$D \in \mathbb{R}^{n_z \times n_z} : d_{ij} = \begin{cases} \frac{1}{\left|z_{o_i}\right|\tau_{r_i} + \tau_{a_i}} & \text{if } i = j, \\ 0 & \text{if } i \neq j. \end{cases} \tag{8.23}$$

The derivatives of f_τ are expressed below in terms of the diagonal matrix D.

$$\begin{aligned} \frac{\partial f_\tau}{\partial z} &= \frac{M}{n_z \left\| z - z^P \right\|_{\mathrm{BDF}}} D^2 [z - z^P] && (8.24) \\ \frac{\partial f_\tau}{\partial h} &= \frac{-M}{n_z \left\| z - z^P \right\|_{\mathrm{BDF}}} [z - z^P] D^2 \dot{z}_o && (8.25) \end{aligned}$$

All the information needed to evaluate these terms is available at each step in the iteration. The denominators defined in (8.24–8.25) are guaranteed to be nonzero at the solution of the system because $\left\| z - z^P \right\|_{\mathrm{BDF}} = (1 - \epsilon)/M$. This ensures that the last row of J is nonzero at the solution. Another advantage obtained when the additional row and column are excluded from the factored portion of the Jacobian matrix is that the elements of these vectors can be updated at every step of the modified Newton iteration.

Initial guesses for $z(h_{\mathrm{est}})$ are provided from the second order Taylor series evaluated at h_{est}. If the iteration fails to converge, the solution of the system is attempted again at $.5h_{\mathrm{est}}$. After two failures, we revert to a standard corrector iteration until a feasible, not optimal, step size is determined.

After successful completion of the integration step, we verify that the interpolation of the calculated solution satisfies the BDF approximation of the model equations. We select a time $h \le h_{\mathrm{opt}}$, such that h is not so small that it requires refactorization of the corrector iteration matrix. We check that the truncation error at this step size is smaller. While this does not guarantee that we have determined the smallest value of h that satisfies (8.16), it verifies that the first interpolated solution approximates the computed solution of the BDF approximation of the model equations (8.15) at the selected time.

The iteration matrix G employed during the modified Newton iteration is refactored according to the same heuristics used to decide whether to reevaluate the corrector iteration matrix in response to a step size change. Therefore, the method proposed will obtain the desired initial step size in the same number or fewer matrix factorizations than would be achieved by simply starting the integrator with the initial guess provided by our estimator, as long as the initial guess h_{est} is close enough to h_{opt} to permit the Newton iteration to converge. If h_{est} is slightly larger than h_{opt}, we obtain the advantage that the second integration step can be taken at the same step size by calculating h_{opt}. If h_{est} is slightly smaller than h_{opt}, a larger step can be employed. Since superlinear convergence of the corrector is achieved, the optimal initial step size is determined with little additional effort. The performance of the method is discussed in section 8.9.

8.8 Implementation within DSL48S

The algorithm described in the preceding sections has been implemented within the DAE code DSL48S (Feehery *et al.*, 1997) — a code derived from DASSL (Petzold, 1982a) that has been designed for large unstructured sparse systems of DAEs, employing the MA48 (Harwell, 1993) linear algebra routines. The code automatically scales the corrector iteration matrix to reflect the error norm employed and minimize the condition number of the resulting corrector iteration matrix (Allgor and Barton, 1997a). In addition, DSL48S employs an efficient method for the integration of the DAE with its associated sensitivity equations. The code either uses a user-supplied routine to evaluate the vector $u_o du/dt + \partial f/\partial t$ required by the algebraic derivative calculation or it determines these using finite differences. If sensitivity equations are integrated as well, then DSL48S either employs the user-supplied routine that provides $\partial^2 f/\partial \dot{x} \partial t \; \partial \dot{x}/\partial p + \partial^2 f/\partial y \partial t \; \partial y/\partial p + \partial^2 f/\partial p \partial t$ evaluated at t_o to determine the derivatives of the algebraic sensitivities, or it determines them using finite differences. All of the other information required to implement the algorithm is readily available within the previous implementation of the code as since the Jacobian is required for integration (DSL48S permits the use of a mixed analytic and numerical Jacobian).

A robust and efficient implementation of the method described in the previous sections requires that certain 'special' cases are identified and dealt with appropriately. First, the value for h_{est} must be provided in cases when $\|\ddot{z}_o\|_{\text{BDF}} = 0$. Two cases are considered depending on whether $\|\dot{z}_o\|_{\text{BDF}} = 0$. If $\|\dot{z}_o\|_{\text{BDF}} \neq 0$, then the following estimate developed by Shampine (1987) is employed:

$$h_{\text{est}} = \frac{(1-\epsilon)}{5\,\|\dot{z}\|_{\text{BDF}}} \tag{8.26}$$

On the other hand, if $\|\dot{z}_o\|_{\text{BDF}} = 0$, then the code defaults to a fraction of the requested initial output length. The difficulty with the implementation of this scheme is to determine when the norms are close enough to zero to be considered zero. We check to see whether $\|\dot{z}_o\|_{\text{BDF}}\, t^{out} / \|z_o\|_{\text{BDF}}$ is small in order to relate the norm to the scale of the problem. Although this scheme may take a conservative initial step size in the case when the system is sitting at steady state, or when only the second derivatives are zero, we feel it is better to take a conservative approach rather than attempt to take the maximum size step that the code will allow. Recognize that if the system is truly operating at steady state, then the augmented system of equations will not have a solution because (8.16) cannot be solved since the prediction is the exact solution of the system.

The efficiency of the iteration is affected by the criteria that are used to determine whether the augmented system of equations (8.15–8.16) is converged. Obviously, the convergence of the variables z^C of the DAE must adhere to the same criteria used for a typical integration step. Since this criteria is based on the size of the updates to the variable values, it will be difficult to satisfy this criteria unless the step size is no longer changing by an appreciable amount. However a slightly smaller initial step size, one that leads to a negative residual in (8.16), is acceptable if the magnitude of the negative residual is close enough to zero; this is analogous to choosing a value of ϵ that is slightly larger than 1/2. These facts indicate that efficiency advantages may be obtained by fixing the step size and merely converging the variable values once (8.16) obtains a negative value close enough to zero. In general, such a strategy is not appropriate to implement within Newton's method because updates for all variables are determined. However, since (8.16) is the last equation in the system and h_o is the last variable, the update to h_o can be set to zero before the back substitution on the rest of the matrix is performed. This is particularly easy to implement in our augmented system since the last row and column are treated separately. As shown in appendix B, the updates to the variables in the absence of a change in h_o are given by v_1. The other advantage to this strategy is that the derivatives of f_τ with respect to both z^C and h often contain significant contributions from numerical error in the evaluation of $z^C - z^P$ which means that the step size may continue to change by small amounts even when its value has essentially converged. By fixing the step size once it is near the answer, these small changes to the step size (possibly caused by numerical error in the derivative expressions) cannot deteriorate the convergence rate of the DAE variables.

The Newton iteration has been modified slightly to improve the convergence when h_{est} is a poor initial guess for h_{opt}. First, every time h is changed by a substantial factor, or on the initial Newton step, only the DAE variables are updated in order to get a more accurate value for f_τ and to be able to evaluate the derivatives of f_τ. This is a tailored recovery strategy from the guaranteed numerical singularity of the Jacobian on the first Newton step that occurs because $z^P = z^{C^{(0)}}$. This allows the Newton step to update the values of the DAE variables on the first step and determine the convergence rate of the Newton iteration. Furthermore, large changes to h are not permitted on a given Newton step; h is not permitted to change by more than an order of magnitude on any given step. If such a large change is indicated, h is changed by an order of magnitude, and the variable values z are determined by the predictor polynomial for a step of this length. This strategy has improved the convergence of the method in situations where h_{est} provides a poor estimate for h_{opt}. On most problems, the largest initial step length that will satisfy the error criteria is desired because the relative size of the contributions of the numerical rounding error to the variable updates will be smaller. Since the error in the approximation of the derivatives is being controlled by the truncation error criterion, the largest step that satisfies the truncation error check should approximate the derivatives to the desired accuracy.

Finally, cases in which the addition of (8.16) to the DAE system leads to a singular system must be handled. These cases arise whenever $z^C - z^P = 0$, so the last row of the matrix becomes zero. Since this always happens on the first Newton step, the tailored recovery strategy mentioned above is employed. However, singularity of this matrix may occur on other steps as well. Whenever the pivot corresponding to h becomes too small, h is doubled (in an attempt to avoid situations where the predictor is extremely accurate), and a standard integration step is attempted at this step length.

8.9 Computational Performance

The computational performance of the algorithm is reported for a set of hybrid discrete/continuous simulation problems. These examples show the benefits of this technique in terms of both an increase in the initial step length and a reduction in the number of Jacobian factorizations and residual evaluations that are required for the overall simulation.

First, the technique is demonstrated for a classic discrete/continuous simulation, the bouncing ball. When the ball is falling, the equations of motion in a gravity field govern its trajectory; these equations define a system of ordinary differential equations. When the ball hits the ground, the ball rebounds with a fraction of the vertical speed at which it contacted the ground according to the coefficient of restitution. The method of Park and Barton (1996) that is used to locate discontinuities during the simulation introduces algebraic variables and equations to the model that represent discontinuity functions. In the case of the bouncing ball, two discontinuity functions are added to the model to identify when the ball hits the ground. The first indicates whether the ball is touching the ground (the center of the ball with diameter .1m is touching if $y \leq .05$); the second ensures that $v_y < 0$ (i.e., the ball is falling). The equations representing the index 1 DAE model of the system are:

$$f = \begin{bmatrix} \dot{x} - v_x \\ \dot{y} - v_y \\ \dot{v}_x \\ \dot{v}_y + 9.81 \\ d_1 + y - .05 \\ d_2 + v_y \end{bmatrix} = 0 \tag{8.27}$$

where x and y represent the position of the center of the ball, and v_x and v_y represent the velocities in each coordinate direction. Initial conditions of $v_x = 1$, $v_y = 0$, $x = 0$, $y = 100$ are specified.

This example demonstrates the advantage of determining the derivatives of the algebraic variables before starting the integration code. The optimal step size h_{opt} (i.e., the step size that satisfies (8.15–8.16)) is calculated with and without the derivatives for the algebraic variables; we denote these as h^{w}_{opt} and h^{wo}_{opt} respectively. When the derivatives of the algebraic variables are not known, a zero order approximation for the algebraic variables is employed for the predictor.

The consistent initialization calculation yields $z_o = [v_{x_o}, v_{y_o}, x_o, y_o, d_{1_o}, d_{2_o}] = [1, 0, 0, 100, 99.95, 0]$ and $[\dot{v}_{x_o}, \dot{v}_{y_o}, \dot{x}_o, \dot{y}_o] = [0, -9.81, 1, 0]$. The derivatives of the algebraic variables are determined by solving (8.5). This yields $[\ddot{v}_{x_o}, \ddot{v}_{y_o}, \ddot{x}_o, \ddot{y}_o, \dot{d}_{1_o}, \dot{d}_{2_o}] = [0, 0, 0, -9.81, 0, 9.81]$. A value of h_{est} is determined from the second derivatives of the differential variables given absolute and relative error tolerances for the variables of 10^{-5}:

$$h_{\text{est}} = \sqrt{\frac{1}{||[0, 0, 0, -9.81]||_{\text{BDF}}}} = \sqrt{\frac{1}{\sqrt{1/4(9.81/.00101)^2}}} = .01435 \qquad (8.28)$$

We employ h_{est} as the initial guess for the solution of (8.15–8.16) when calculating both h^{w}_{opt} and h^{wo}_{opt}.

We examine the solution of (8.15–8.16) with and without the derivatives of the algebraic variables. Both h^{w}_{opt} and h^{wo}_{opt} solve (8.15–8.16); the values differ due to the way that the solution is approximated at the initial time. When we include the derivatives of the algebraic variables, $\dot{z}_o = [0, -9.81, 1, 0, 0, 9.81]$ and $h^{w}_{\text{opt}} = 9.431 \times 10^{-3}$. If we do not employ the derivatives of the algebraic variables, $\dot{z}_o = [0, -9.81, 1, 0, 0, 0]$ and $h^{wo}_{\text{opt}} = 1.248 \times 10^{-6}$. These step sizes differ by a factor of about 7500, requiring almost 13 additional steps, doubling the step size at each step, for h^{wo}_{opt} to achieve the magnitude of h^{w}_{opt} calculated when the algebraic derivatives were provided. Since the heuristics within DSL48S refactor the iteration matrix every time the step size is doubled (unless the order is also increased), the cost of these additional factorizations will be significant on large models. The cost required to determine these derivatives is comparable to the cost of one factorization of the iteration matrix. Note that the calculation of the algebraic derivatives also provided $\ddot{z}_o$ which was used to calculate h_{est}, the initial guess for h_{opt}.

Test Problem Name	Number of Events	Performance Measures				
		Jacobian Factorizations	Int. Steps	Residual Evals.	Convergence Failures	Error Failures
Bouncing Ball	7	203	245	388	0	6
Safety Valve	12	165	274	465	0	12
Flash	11	202	586	1304	10	28
Valve	5	52	201	396	0	6
Event/Simulate2	18	192	583	1140	0	9
Event/Simulate4	10	166	396	793	0	25
Series Reactions	1	14	73	145	0	0

Table 8.1: Performance of integration code on combined simulation test problems using the initial step length heuristics employed by DASSL.

Results are presented to compare the performance of the initialization procedure on a host of test problems using the default implementation contained in DASSL (see table 8.9) and the optimal initial step length calculation proposed in this work (see table 8.9). For each problem, the approach just presented for the selection of the initial step size is compared with the heuristic implemented within DASSL; DASSL's heuristic estimates the initial step length as either a fraction of the length of first output interval or according to the inverse of the norm of the variable derivatives. Note that the heuristics employed within DASSL permit the step size to be doubled and the order increased at the completion of each successful step in the initial phase of the integration. In contrast, the method used here employs the conservative step size adjustment procedures employed throughout the code at the completion of the initial step.

8.10 Conclusions

The statistics presented in the preceding section demonstrate that the method used to calculate the initial step size improves both the reliability and efficiency of the BDF integration code in the initial phase of the integration. This applies to each initial value problem encountered during the solution of a combined simulation experiment. The increase in the efficiency of the method stems from both the availability of the derivatives of the algebraic variables on the first step and the simultaneous calculation

Test Problem Name	Number of Events	Performance Measures Jacobian Factorizations	 Int. Steps	 Residual Evals.	 Convergence Failures	 Error Failures
Bouncing Ball	7	126	168	285	0	0
Safety Valve	12	132	250	404	0	0
Flash	11	195	603	1314	0	14
Valve	5	43	197	414	0	10
Event/Simulate2	18	82	470	947	0	0
Event/Simulate4	10	154	404	797	0	11
Series Reactions	1	9	67	134	0	0

Table 8.2: Performance of integration code on combined simulation test problems using the optimal initial step length calculation.

of the variable values and the initial step length during the first integration step.

Using the derivatives of the algebraic variables at the initial time improves the accuracy of the prediction during the first integration step. Without these derivatives the initial step length will be restricted to much smaller values. In fact, if the first order terms in the Taylor series for the algebraic variables dominate the higher order terms, then the initial step size cannot be greater than the point at which the norm of the first order terms exceeds the allowable error tolerance (i.e., $\tilde{h}_o \leq (n_y + n_x)M/(n_y \, \|\dot{y}\|_{\text{BDF}}))$. The value $\tilde{h}_o$ approximates the largest step size that could succeed on the initial step if the $\dot{y}_o$ are not determined. Since the derivatives of the algebraic variables can be calculated inexpensively, the benefits appear clear. Determining these values allows the size of the initial step length to be governed by the second order terms in the Taylor series. This additional calculation improves the performance of the integration of DAEs, distinguishing this method from those applied to ODEs. In addition, the algebraic derivative calculation provides the second derivatives of the differential variables $\ddot{x}_o$ which can be used to estimate the length of the optimal initial step length.

The second derivatives of the differential variables provide information that can be employed to estimate an initial step size that maintains the validity of the error estimate but is on scale for the problem. The method presented establishes criteria that define the optimal initial step length. We have demonstrated that a step satisfying these criteria can be found by augmenting the system of equations solved during the corrector iteration. The augmented system of equations can be solved using the same corrector iteration matrix. Whenever a good initial estimate of the optimal step size is calculated by our estimation procedure, the optimal initial step size can be determined without any additional factorizations of the corrector iteration matrix. The solution statistics for the example problems demonstrate the improvements of the efficiency of the solution procedure. In addition, the step size selection procedure employed during the initial phase of the integration is more conservative and leads to fewer convergence and error test failures, yet it remains more efficient.

Since this method improves both the efficiency and reliability of the code in the initial phase of the integration, it can provide significant benefits for the hybrid discrete/continuous simulation of large models with frequent discontinuities. The method is ideally implemented within combined simulation environments where the required derivative information is available.

Chapter 9

Mixed-Integer Dynamic Optimization

This chapter presents some preliminary results on how the decomposition approach for the batch process development problem introduced in chapter 2 extends to a more general class of mixed-integer dynamic optimization problems. We define *mixed-time-invariant-integer dynamic optimization* as the class of problems for which the decomposition strategy applies, and demonstrate that simple extensions of mixed-integer nonlinear programming (MINLP) techniques are doomed to failure on this class of problems. On the other hand, our approach combines dynamic optimization with insight based targeting techniques to decompose the optimization into subproblems providing rigorous upper and lower bounds on the objective. This approach has the potential to eliminate total enumeration of the discrete space, assures termination in a finite number of iterations, and yields a rigorous bound on the distance between the solution found and the global solution.

9.1 Introduction

Many problems in process design and operation require the optimal selection of quantities that vary over time. When a mathematical model of the process is available, these quantities may be calculated using dynamic optimization; in fact, several researchers in the chemical engineering community have developed algorithms for the optimization of large-scale dynamic systems (Cuthrell and Biegler, 1987; Vassiliadis, 1993; Feehery and Barton, 1996a). However, many problems also contain discrete quantities or decisions that cannot be described using purely continuous dynamic models of the system. The growing recognition of the importance of discrete/continuous (or *hybrid*) dynamic systems to the chemical industry has recently motivated the development of appropriate simulators (Barton and Park, 1997). Similarly, the optimization of hybrid dynamic systems cannot always be performed using purely continuous formulations. This motivates new algorithms capable of handling classes of mixed-integer dynamic optimization (MIDO) problems.

Recently, dynamic optimization of large scale continuous systems has been demonstrated (Charalambides *et al.*, 1995b), and dynamic optimization capabilities have even been embedded in process simulators such as ABACUSS. However, limited progress has been made that addresses dynamic problems coupled with discrete decisions. Charalambides *et al.* (1993) formulate 'batch process synthesis' as a multistage mixed-integer dynamic optimization problem, but no solution procedures have been reported. Mohideen *et al.* (1996) consider design and control in the presence of uncertainty, formulating the problem as a stochastic mixed-integer optimal control problem. This problem is transformed into a finite dimensional MINLP through discretization of the time domain with orthogonal collocation on finite elements. However, the nonconvexities inherent in this problem are not discussed, so the application of traditional MINLP algorithms to this problem is likely to reduce to an ad hoc improvement strategy that may prune the optimal discrete alternative (Sahinidis and Grossmann, 1991; Bagajewicz and Manousiouthakis, 1991).

In contrast, we present a decomposition approach to MIDO that is capable of providing rigorous bounds on the global solution in spite of the nonconvexities inherent in the variational subproblems. In addition, this decomposition is the first that permits either collocation or numerical integration based strategies to be used for the variational subproblems. In the following sections, we formally define the MIDO algorithm and the class of problems it addresses. Further, we demonstrate how the required subproblems can be derived and solved on a relatively simple batch process development example.

9.2 Problem Scope

We consider the class of mixed-integer dynamic optimization problems that conform to the following formulation:

$$\min_{u(t),v,y,t_f} \left\{ \sum_k \phi_k(x_k(t_k^f), u_k(t_k^f), v, y, t_k^f) + \sum_k \int_{t_k^0}^{t_k^f} L_k(x_k(t), u_k(t), v, y, t) dt \right\} \tag{9.1}$$

Subject to:

$$f_k(x_k(t), \dot{x}_k(t), u_k(t), v, y, t) = 0 \qquad \forall\, k,\ t \in [t_k^0, t_k^f] \tag{9.2}$$

$$g_k(x_k(t), \dot{x}_k(t), u_k(t), v, y, t) \leq 0 \qquad \forall\, k,\ t \in [t_k^0, t_k^f] \tag{9.3}$$

$$h(v, y, t) \leq 0 \tag{9.4}$$

$$\alpha_{kp}(x_k(t_p), \dot{x}_k(t_p), u_k(t_p), v, y, t_p) \leq 0 \qquad \forall\, k,\ p \in \{0, n_{p_k}\} \tag{9.5}$$

where

$$x_k \in X_k \subseteq \mathbb{R}^{n_x} \quad u_k \in U_k \subset \mathbb{R}^{n_{u_k}} \quad \forall k$$

$$u \in \bigcup_k U_k = U \subseteq \mathbb{R}^{n_u} \quad v \in V \subseteq \mathbb{R}^{n_v} \quad y \in Y = \{0,1\}^{n_y}$$

$$f_k : X_k \times \mathbb{R}^{n_{x_k}} \times U_k \times V \times [0,1]^{n_y} \times \mathbb{R} \to \mathbb{R}^{n_{x_k}}$$

$$g_k : X_k \times \mathbb{R}^{n_{x_k}} \times U_k \times V \times [0,1]^{n_y} \times \mathbb{R} \to \mathbb{R}^{n_{g_k}}$$

$$h : V \times [0,1]^{n_y} \times \mathbb{R} \to \mathbb{R}^{n_h}$$

$$\alpha_{kp} : X_k \times \mathbb{R}^{n_{x_k}} \times U_k \times V \times [0,1]^{n_y} \times \mathbb{R} \to \mathbb{R}^{n_{k_p}}$$

and $x_k(t)$ are the continuous variables describing the state of the dynamic system k, $u_k(t)$ are continuous controls whose optimal time variations on the interval $[t_k^0, t_k^f]$ are required, v are continuous time invariant parameters whose optimal values are also required, y are a special set of time invariant parameters that can only take binary values, and t_k^f is a special continuous time invariant parameter known as the final time of system k. This formulation allows for n_k dynamic models that are coupled by the time invariant parameters v and y. It is the presence of the binary time invariant parameters y that distinguishes formulation (9.1–9.5) from other recent quite general dynamic optimization formulations (Vassiliadis *et al.*, 1994). We conjecture the existence of a more general class of problems that also contain binary controls (i.e., functions whose time variation is restricted to take 0-1 values) but will only consider the class (9.1–9.5). Hence, to coin a term, (9.1–9.5) might be called a *mixed time invariant integer dynamic optimization.*

The constraints (9.2–9.5) warrant some explanation. Equations (9.2) represent a general set of differential-algebraic equations (DAEs) describing the kth dynamic system; each dynamic model k can only interact with another dynamic model $k' \neq k$ through the time invariant parameters. As such, (9.2) will include a lumped dynamic model of the system in question coupled with any path equality constraints that system k must satisfy; the number of controls that remain as decision variables in the optimization is reduced by each path equality constraint added to the formulation. Note that for any admissible realization of the $\{u(t), v, y, t_f\}$ (one that satisfies the logical constraints (9.4) and produces a solvable DAE) the choice of which degrees of freedom to designate as controls $u(t)$ and the presence of path constraints may have a profound influence on the *differential index* (Brenan *et al.*, 1996) of (9.2). For practical purposes, we will further assume that, while (9.2) may have arbitrary index, the index is time invariant and can be correctly determined using structural criteria. Hence, the method of dummy derivatives may be used either for numerical solution of the initial value problems (IVPs) in (9.2) (Mattsson and Söderlind, 1993; Feehery and Barton, 1996a), or to derive an equivalent index-1 discretization of (9.2) via collocation (Feehery and Barton, 1995). Here, we emphasize that the differential index of the model solved may be a function of y, but that the index must remain time invariant for any integer realization of y. For example, the following system is index-2 if $y_1 = y_2$ and index-1 otherwise:

$$\begin{aligned} \dot{x}_1(t) &= -x_1(t) + x_2(t) \\ x_1(t) + (y_1 - y_2)x_2(t) &= u(t) \\ y_1 + y_2 &\leq 1 \end{aligned} \tag{9.6}$$

Inequalities (9.3) represent a general set of path inequality constraints that must be satisfied by a solution of the optimization. Feehery and Barton (1996b) discuss an algorithmic approach to the solution of dynamic optimizations containing such path constraints. This approach will invoke further assumptions concerning inequalities (9.3), arising from the need to couple (9.2) with any active members of (9.3) during the solution process. Specifically, we require that the coupled system formed when some of the constraints (9.3) are active and some of the controls are treated as state variables remains solvable for the selected partition of the control variables. Constraints placed on the dynamic model at specific times, such as initial conditions or final time requirements, are represented by (9.5). In addition, (9.4) defines constraints that coordinate the operation of the n_k different dynamic models through the time invariant integer (y) and continuous (v) parameters. Note that models that cannot be decoupled through the use of time invariant parameters can be represented within this formulation by permitting only one dynamic model (i.e., $n_k = 1$).

9.3 Applying MINLP algorithms

The development of our approach for mixed-integer dynamic optimization proceeds from an analogy with algorithmic approaches to MINLP. An excellent review and discussion of MINLP algorithms is given by Floudas (1995). First, we examine the applicability of two popular and general approaches used for MINLP problems to the MIDO problem. We discuss both Branch and Bound approaches, analogous to those used for MILP problems, and decomposition approaches such as the Generalized Benders Decomposition (GBD) (Geoffrion, 1972) and the Outer Approximation

Method (OA) (Duran and Grossmann, 1986) and its variants. The problems that may be encountered when extending either of these techniques to the MIDO problem are discussed, which leads us to pursue an alternative decomposition approach for mixed integer dynamic optimization based on domain specific knowledge.

A Branch and Bound approach to MIDO requires the existence of a continuous relaxation to problem (9.1–9.5), and the ability to solve this relaxation to global optimality. The required relaxation poses both theoretical and practical problems. For example, problems for which the DAE (9.2) is solvable for integral values of y but is not solvable for one or more values of $y \in (0, 1)$ can be constructed quite easily. The linear time varying DAE system (9.7) coupled with the logical point constraint (9.8) serves as a pathological example:

$$\begin{bmatrix} -2y_1 t & 2y_2 t^2 \\ -1 & 2y_1 t \end{bmatrix} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{9.7}$$

$$y_1 + y_2 \leq 1 \tag{9.8}$$

Brenan *et al.* (1996) show that the DAE (9.7) which arises when $y = [.5, .5]^T$ has the solution $x = \phi(t)[t, 1]^T$ for any function $\phi(t)$, demonstrating that the solution is not unique. However, (9.7) is solvable for any integer realization of y that satisfies (9.8). In addition, (9.7) forms an index 2 system at $t = 0$ for certain integer realizations of y, and is index 1 at other times; while this does not relate to the solvability of (9.7), it may cause practical difficulties for any integration procedure.

Similarly, the index of (9.2) can vary locally for y in the interval $(0, 1)$ even though the index may be well defined according to structural criteria for integral values of y; for example, see (9.6). Local variations in the index create severe problems for current general purpose approaches to the numerical solution of high index DAEs (Feehery and Barton, 1996a).

More importantly, even if we assume that a valid continuous relaxation exists, any but the simplest dynamic optimization problems exhibit multiple local optima almost pathologically, as shown by Banga and Sieder (1995). Furthermore, no current techniques can solve a general dynamic optimization to *guaranteed* global optimality (disregarding the prohibitive computation a global optimal control would require), and there are no indications that such a technique will be developed in the near future. Since we cannot guarantee that a relaxation of (9.1–9.5) can be solved to global optimality, relaxed solutions cannot serve as valid lower bounds for implicit enumeration of the Branch and Bound tree. Therefore, a Branch and Bound approach to MIDO is doomed to explicitly enumerate the Branch and Bound tree. In contrast, the decomposition approach that we propose does not require a global solution of the dynamic optimization, yet it still offers the potential to avoid total enumeration of the discrete space.

Decomposition approaches for MINLP are based on the idea that sequences of rigorous upper (nonincreasing) and rigorous lower (nondecreasing) bounds can be derived that will converge within a finite number of iterations. Convergence occurs when the upper and lower bounds approach to within the desired tolerance, or when all the discrete alternatives lying beneath the current upper bound have been enumerated. The different decomposition algorithms are distinguished by the way in which these sequences are generated and by the properties required to ensure validity of the bounds. For example, basic GBD places strict conditions on the functions appearing in the MINLP in order to derive an equivalent dual representation of the problem; relaxations of the dual are then used to generate a sequence of valid nondecreasing lower bounds for classes of MINLPs adhering to these restrictions. For all other decomposition approaches, similar restrictions are placed on the type of models to which the algorithm can be applied successfully.

The upper bound in a decomposition approach is calculated in a similar manner in all cases: the binary variables y are fixed to integer values, reducing the MINLP to a NLP that can then be solved to yield a rigorous upper bound on the solution; the upper bound is valid even if the global solution to the NLP is not found. When the y are fixed to integer values, the MIDO (9.1–9.5) can be viewed as a NLP since an equivalence can be established between the classical necessary conditions for optimality of a continuous dynamic optimization (Bryson and Ho, 1975) and the local solution of an NLP in the context of either control parameterization (Kraft, 1985) or collocation (Logsdon and Biegler, 1989). However, in general, this NLP will not possess the theoretical properties required for successful application of MINLP decomposition techniques; the global optimum of the NLP must be guaranteed *and* the Primal must permit the derivation of valid support constraints for the Master problem (Floudas, 1995). In particular, it is important to stress that obtaining the global optimum of the dynamic optimization is not sufficient for the application of OA and GBD techniques (Sahinidis and Grossmann, 1991; Bagajewicz and Manousiouthakis, 1991). These theoretical barriers have motivated this investigation of an alternative decomposition approach that does not require that these properties are maintained by the primal. In our approach, sequences of nonincreasing upper bounds and nondecreasing lower bounds are retained. In addition, we introduce the notion of a primal bounding model to permit the method to exploit either the global solution of the dynamic optimization problem or tighter convex underestimators of the primal than those furnished by a screening model.

9.4 Decomposition Approach to MIDO

We propose a decomposition approach in which the lower bounding model does not depend on the solution properties of the continuous optimization problem. In fact, the lower bounding model is derived from domain specific knowledge gathered from physical laws and engineering insight. The algorithm assumes the existence of the following subproblems:

Master Problem which is the solution to a so-called *screening model.* This model can be solved to guaranteed global optimality to yield a rigorous lower bound on the solution to the MIDO. The model is derived from domain specific knowledge.

Primal Problem which is the solution of the continuous dynamic optimization resulting from fixing y in (9.1–9.5) to an admissible integer realization. This yields a rigorous upper bound on the solution to the MIDO.

Primal Bounding Problem which provides a tighter lower bound on the solution to the primal problem for a fixed realization of y than that provided by the Master. Note that this subproblem, unlike the other two, is not absolutely necessary, but its existence can improve our estimate of the quality of the solution obtained.

We denote the solution of the master and primal problems at each iteration as z_k^M and z_k^P respectively, and define $\hat{z}_k^P$ as a lower bound on the solution of the primal at each iteration. Obviously after every iteration of the primal subproblem $z_k^M \leq \hat{z}_k^P \leq z_k^P$. Limiting cases are observed when one of these two inequalities is always satisfied with equality, in which case we have either found the global optimum of the primal, or we have no tighter lower bound for the primal than the one provided by the master problem. The following algorithm simplifies in these two limits. We also denote the lower bound on the global solution by LBD and the upper bound on the global solution as UBD and choose to update both at every iteration. The current solution of the master problem is used to terminate the iteration sequence. A flowchart of the following algorithm is shown in figure 9-1:

1. Initialize:
   (a) iteration counter $k = 1$
   (b) $LBD = -\infty$, $UBD = \infty$
2. Solve Master Problem.
   (a) Obtain z_k^M.
   (b) $LBD = \min_{k'<k}(z_k^M, \hat{z}_{k'}^P)$
3. Terminate if $z_k^M > UBD$ or if the Master Problem was infeasible.
   (a) The distance from the best solution found to the global minimum is known to be less than $UBD - LBD$.
   (b) The global solution is described by one of the discrete alternatives that has been examined ($y \in \{y_{k'} : \ \forall k' < k\}$).

4. Solve the Primal and Primal Bounding Problems.

   (a) Obtain z_k^P and $\hat{z}_k^P$. If the Primal Bounding Problem does not exist, then the lower bound for the primal is assigned to the solution of the master: $\hat{z}_k^P = z_k^M$.

   (b) $UBD = \min(UBD, z_k^P)$

5. Add to the Master Problem an integer cut that excludes y_k, and any constraints that can be derived rigorously from the primal solution.

6. $k = k + 1$. Return to step 2.

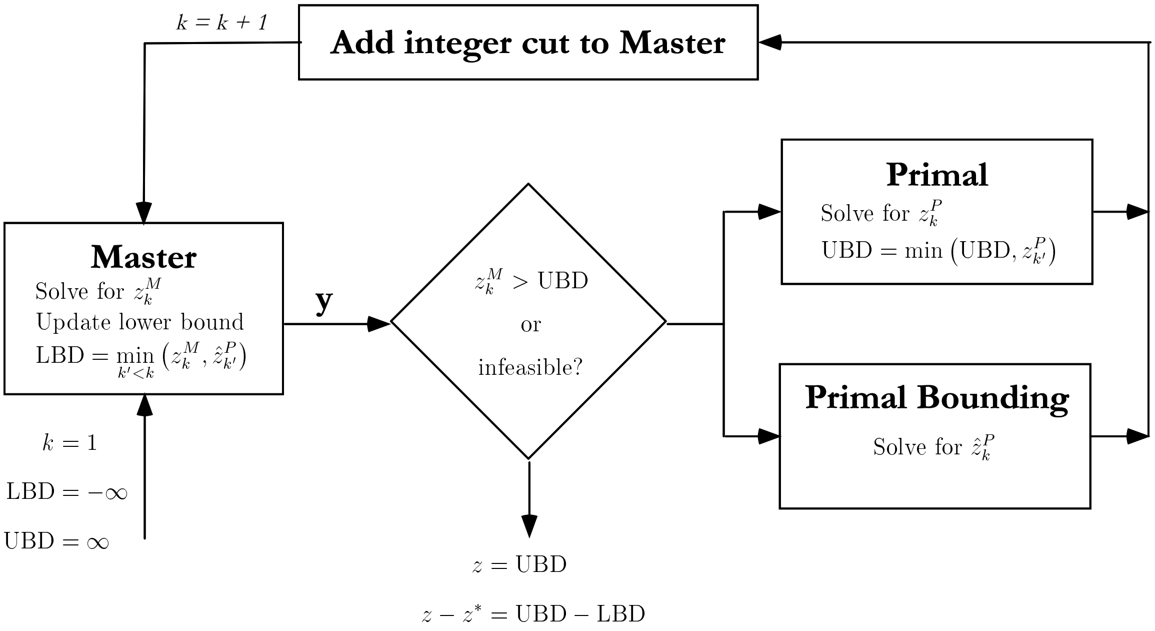

Figure 9-1: Flowchart of the MIDO decomposition algorithm.

Figure 9-2 depicts a sequence of iterates that could be achieved from the algorithm, illustrating both the termination criterion and the bound on the distance to the global solution. Below we prove that the optimal discrete alternative has been examined and explain the role that the primal bounding model plays in determining the bound on the distance from the solution obtained to the global optimum.

First, we prove that on termination the optimal discrete alternative has been examined by showing that the unexplored discrete alternatives must result in solutions

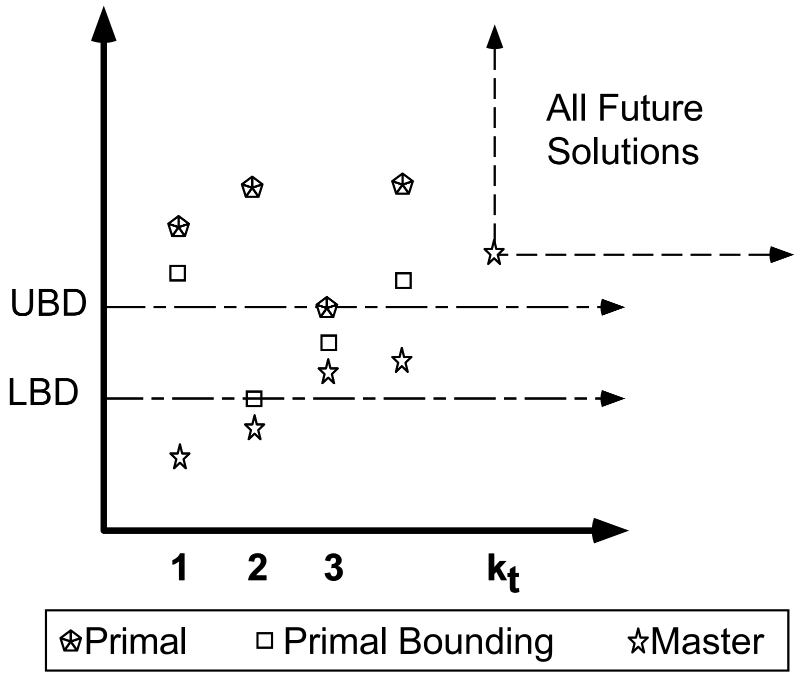

Figure 9-2: Sequence of subproblem solutions that could be obtained from the MIDO decomposition algorithm.

with a higher objective value than UBD. The Master Problem is valid only if it provides a rigorous lower bound on the corresponding Primal Problem, so the following holds:

$$z_k^M \leq z_k^P \quad \forall k \tag{9.9}$$

Introducing an integer cut at each iteration of the Master Problem generates a series of steadily increasing solutions.

$$z_k^M \leq z_{k+1}^M \quad \forall k \tag{9.10}$$

Upon termination of the iteration sequence, we know that the Master Problem is either infeasible or that the solution of the Master is greater than the current upper bound UBD. If the Master Problem is infeasible, all of the remaining discrete alternatives are infeasible and need not be examined. If the solution of the Master is greater than the current upper bound, (9.9–9.10) show that any future iterations will result in solutions that are greater than the current upper bound. This proves that iteration technique is capable of avoiding total enumeration of the discrete alternatives, and that the discrete alternative leading to the global solution has been investigated.

Next we verify that we have obtained a bound on the distance to the global solution. We recognize that the global solution must be greater than the minimum of the primal lower bounds $LBD \geq \min_k \hat{z}_k^P$. Note that this contrasts with conventional MINLP algorithms in which the solution of the Master problem always provides the lower bound. In conventional MINLP algorithms the global solution of the Primal problem is guaranteed, so the lower bound can be updated after each solution of the Master problem. However, for the MIDO problem the solution of the Primal is not guaranteed to provide the global optimum, so the lower bound can only be updated if the solution of the Master is guaranteed to be less than the global optimum of all of the previously examined Primal problems. Since the solution of the Primal Bounding model provides a rigorous lower bound on the solution of the Primal problem, the lower bound can be updated after the solution of the Master problem as long as the solution of the Master is not greater than any of the solutions of the Primal bounding model found so far. Figure 9-2 shows that on the second iteration the lower bound was updated after the solution of the Master problem, since $z_2^M < \hat{z}_1^P$. However, after the solution of the third Master problem, LBD cannot remain at the value given by the $\hat{z}_2^P$ because the possibility exists that a solution of the Primal problem with value less than z_3^M exists. The least upper bound is simply UBD, the infimum of the solutions of the primal subproblems. Therefore the distance between the solution at termination and the tightest bound we have obtained on the global solution is given by $UBD - LBD$.

Since z_k^M is forced to be nondecreasing at each step (through the introduction of the integer cuts), and there are a finite number of integral realizations of y_k, the algorithm will terminate after a finite number of iterations. Depending on how tight the screening model is, this property has the potential to avoid enumeration of the entire discrete decision space.

9.5 Casting Batch Process Development as a MIDO

This section demonstrates that the batch process development problem can be formulated as a mixed time invariant integer dynamic optimization problem that conforms to (9.1-9.5). For illustration, the batch process development example from chapter 4 is formulated according to (9.1-9.5).

The goal of the MIDO is to select the values for the time invariant parameters and control profiles that minimize the production cost per unit mass of product P using equipment that is available within the existing manufacturing facility. The processing costs are evaluated assuming cyclic steady state for the duration of the campaign, ignoring end effects. We employ simple dynamic models of both the distillation column and the reactor for the purposes of illustration. More complicated dynamic models can be employed within the formulation, but they would make the expression of all the model constraints within this text far more cumbersome. In the following model, time invariant parameters are represented by v and the controls are represented by u. The reactor temperature, the feed rate of reagent, the column reflux ratio, and the positions of the valves governing the flow into the accumulators are treated as the control variables in the optimization. The superscripts on the controls and the time invariant parameters indicate what the particular controls and parameters represent. Note that each task is denoted by the subscript k. This differs slightly from the notation employed in chapter 4 in which the subscript k referred to processing trains. We consider a superstructure with two distillation and reaction tasks, and let k refer to an element taken from the ordered set $K = \{R1, D1, R2, D2\}$, and let K_R and K_D refer to the order subsets of the reaction and distillation tasks. Let inequality (e.g., $k < k'$) and arithmetic operations (e.g., $k - 1$) refer to operations performed with respect to the ordinality of the elements of the set.

We employ time invariant parameters to represent the state of the material entering and leaving each of the tasks. These material states are represented by the tanks surrounding each of the tasks shown in figure 9-3. The mass balance around each of these tanks is enforced by constraints on the time invariant parameters. Figure 9-3 denotes material transfers described by the model equations using solid lines and transfers that occur at the beginning and end of the tasks using dotted or dashed lines; these transfers are represented by point constraints in the formulation. Transfers between these tanks are represented by point constraints in the model.

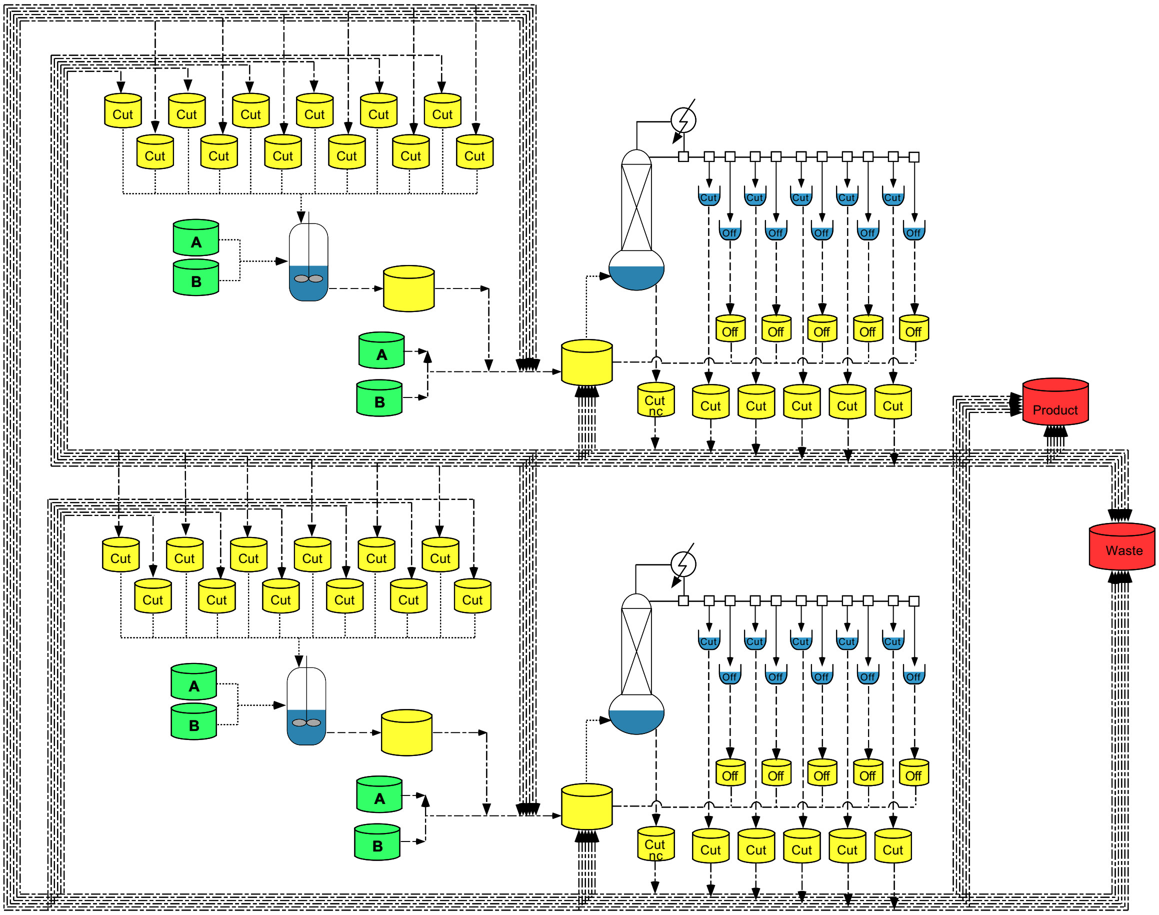

Figure 9-3: The superstructure for the MIDO formulation of the process development example from chapter 4.

9.5.1 Distillation Column Constraints

A simple equilibrium stage model of the batch distillation is employed (Bernot *et al.*, 1990). The model assumes no holdup on the trays, constant pressure, and does not enforce energy balances. All of the material in the column is contained in the liquid of the reboiler. The Wilson activity coefficient model and the extended Antoine vapor pressure model are used to determine the vapor-liquid equilibrium, but have simply been represented below using f^{VLE} which defines the relationship between liquid and vapor composition, pressure, and temperature. We ignore utility costs in this example in order to simplify the distillation model.

The model of the distillation column contains NS equilibrium stages in the tray section, resulting in $NS+1$ stages. The first stage corresponds to the reboiler.[1] The model of the distillation accounts for multiple columns of the same type operating in exactly the same fashion. This permits the columns to be modeled as one larger column operating at a vapor rate that represents the sum of the individual rates.

Reboiler:

$$\frac{dM_{ek}}{dt} = -\frac{V_k^{\mathrm{vapor}}}{u_k^R+1} x_{ek}^D y_k^D \qquad \forall e, k \in K_D \tag{9.11}$$

$$M_k^{\mathrm{total}} = \sum_e M_{ek} \qquad \forall k \in K_D \tag{9.12}$$

$$M_{ek} = M_k^{\mathrm{total}} x_{eks} \qquad \forall e, k \in K_D, s = 1 \tag{9.13}$$

$$V_k^{\mathrm{mol}} = f^{\mathrm{Vol}}(\mathbf{x}_{ks}, T_{ks}, P_k) \qquad \forall\ k \in K_D, s = 1 \tag{9.14}$$

Equilibrium Stages:

$$\mathbf{f}^{\mathrm{VLE}}\left(\mathbf{x}_{ks}, \mathbf{y}_{ks}, T_{ks}, P_k\right) = 0 \qquad \forall\ k \in K_D, s = 1, ns+1 \tag{9.15}$$

$$\sum_e y_{eks} = 1 \qquad \forall\ k \in K_D, s = 1, ns+1 \tag{9.16}$$

Operating Line:

$$\mathbf{x}_{ks} u_k^R = \mathbf{y}_{k,s-1}\left(u_k^R+1\right) - x_k^D \quad \forall\ k \in K_D, s = 2, ns+1 \tag{9.17}$$

[1] Although this goes against the usual numbering convention, we have found that treating the reboiler as the first stage makes it considerably easier to provide a guess for the initial column profile, since the initial profile from a column with fewer stages can be used as the initial guess for the column with more stages.

Condenser:

$$\mathbf{x}_k^D = \mathbf{y}_{ks} \qquad \forall\, k \in K_D, s = ns + 1 \tag{9.18}$$

$$D_{ek} = -\frac{dM_{ek}}{dt} \qquad \forall\, e, k \in K_D \tag{9.19}$$

$$\sum_{c=1}^{nc-1} \left(u_{ck}^{S_{\text{cut}}} + u_{ck}^{S_{\text{off}}} \right) = 1 \qquad \forall\, k \in K_D \tag{9.20}$$

$$\frac{dM_{cek}^{\text{cut}}}{dt} = D_{ek} u_{ck}^{S_{\text{cut}}} \qquad \forall\, c = 1, nc-1,\ e, k \in K_D \tag{9.21}$$

$$\frac{dM_{cek}^{\text{off}}}{dt} = D_{ek} u_{ck}^{S_{\text{off}}} \qquad \forall\, c = 1, nc-1,\ e, k \in K_D \tag{9.22}$$

Note that the fraction of the distillate fed to each of the cut and off cut accumulators of the distillation column is specified by the control variables $u_{ck}^{S_{\text{cut}}}$ and $u_{ck}^{S_{\text{off}}}$. Since all of the distillate must be sent to the accumulators, (9.20) requires that these controls sum to one. The fraction of the distillate flow reaching the the splitter above each of the accumulators that is sent to the accumulator can be defined as follows for the cuts and off cuts respectively:

$$\text{Split Fraction for Cut } ck = \frac{u_{ck}^{S_{\text{cut}}}}{1 - \sum_{c'=1}^{c-1} \left(u_{c'k}^{S_{\text{cut}}} + u_{c'k}^{S_{\text{off}}} \right)}$$

$$\text{Split Fraction for Off Cut } ck = \frac{u_{ck}^{S_{\text{off}}}}{1 - \sum_{c'=1}^{c} u_{c'k}^{S_{\text{cut}}} - \sum_{c'=1}^{c-1} u_{c'k}^{S_{\text{off}}}}$$

The split fractions for each of the splitters (or the position of the valves directing the flow into the accumulator) are not included as controls in the problem, but can be calculated from $u_{ck}^{S_{\text{cut}}}$ and $u_{ck}^{S_{\text{off}}}$ easily.

The operation of the column requires specification of the initial conditions and any requirements that are placed on the final state of the operation. These constraints follow.

Inequality Path Constraints:

$$y_k^D V_k^{\text{mol}} M_k^{\text{total}} \le \sum_{i \in I_D} \sum_{n=1}^{N_i} y_{ikn}^C n \hat{V}_i \qquad \forall k \in K_D \tag{9.23}$$

$$0 \le u_{ck}^{S_{\text{cut}}} \le 1 \qquad \forall k \in K_D, c = 1, nc - 1 \tag{9.24}$$

$$0 \le u_{ck}^{S_{\text{off}}} \le 1 \qquad \forall k \in K_D, c = 1, nc - 1 \tag{9.25}$$

$$1.5 \le u_k^R \le 15 \qquad \forall k \in K_D \tag{9.26}$$

Final Time Constraints:

$$\sum_e M_{cek}^{\text{cut}} = v_{ck}^{M^{\text{cut}}} \qquad \forall\ c = 1, nc - 1,\ k \in K_D \tag{9.27}$$

$$M_{cek}^{\text{cut}} = v_{ck}^{M^{\text{cut}}} v_{cek}^{X^{\text{cut}}} \qquad \forall\ c = 1, nc - 1,\ e, k \in K_D \tag{9.28}$$

$$\sum_e M_{cek}^{\text{off}} = v_{ck}^{M^{\text{off}}} \qquad \forall\ c = 1, nc - 1,\ k \in K_D \tag{9.29}$$

$$M_{cek}^{\text{off}} = v_{ck}^{M^{\text{off}}} v_{cek}^{X^{\text{off}}} \qquad \forall\ c = 1, nc - 1,\ e, k \in K_D \tag{9.30}$$

$$\sum_e M_{ek} = v_{ck}^{M^{\text{cut}}} \qquad \forall\ c = nc,\ k \in K_D \tag{9.31}$$

$$M_{ek} = v_{ck}^{M^{\text{cut}}} v_{cek}^{X^{\text{cut}}} \qquad \forall\ c = nc,\ e, k \in K_D \tag{9.32}$$

Initial Time Constraints:

$$M_{ek} = v_k^{M^{\text{mix}}} v_{ek}^{X^{\text{mix}}} \qquad \forall\ e, k \in K_D \tag{9.33}$$

$$M_{cek}^{\text{cut}} = 0 \qquad \forall\ c = 1, nc - 1,\ e, k \in K_D \tag{9.34}$$

$$M_{cek}^{\text{off}} = 0 \qquad \forall\ c = 1, nc - 1,\ e, k \in K_D \tag{9.35}$$

9.5.2 Reaction Constraints

The reaction step comprises the set of competing reactions shown below. All of the reactions are first order under the conditions in which the process may be operated;

reactions 1 and 2 are first order in A.

$$\begin{tikzcd}
A+B \arrow[r, "1"] \arrow[d, "2"] & I \arrow[r, "3"] \arrow[d, "4"] & P \\
W_1 & W_2 &
\end{tikzcd}$$

The relative rates of the reactions have been chosen so that they agree with an early study of reaction temperature optimization (Denbigh, 1958) following Arrhenius rate expressions given by the constants in table 4.1.

A simple dynamic model of the reactor is employed. Both the temperature and the rate at which material is charged to the reactor are treated as controls. At the completion of the reaction task the impeller is stopped, and the catalyst settles to the bottom. We assume that the reactions stop at this point. Note that the model of the standard reaction task has been augmented to include the equality path constraints defining the amount of material charged from each of the feed tanks. Several design constraints restrict the operation of the reactor. A molar ratio of solvent to A of at least 15 is required; the components B, W_1, and W_2 all can act as the solvent, and all of the solvent must be charged initially. An excess of B (two times A) is also required. We assume that parallel reactors operate in phase. In this model we assume that reactors only differ in size, so multiple reactors in parallel can be modeled as one larger reactor, simplifying the model of the reaction task.

DAE model of reaction task:

$$\frac{dM_{ck'k}^{R^{\text{in}}}}{dt} = u_{ck'k}^{R^{\text{in}}} \qquad \forall\ c,\ k \in K_R, k' \in K_D \tag{9.36}$$

$$\frac{dM_{ek}^{S^{\text{in}}}}{dt} = u_{ek}^{S^{\text{in}}} \qquad \forall\ e \in E_R\ k \in K_R \tag{9.37}$$

$$\frac{dM_{ek}}{dt} = \sum_{r \in R_k} y_k^{Rxn} \text{rate}_{kr} \nu_{ekr} + u_{ek}^{S^{\text{in}}}$$

$$+\sum_{c}\sum_{k'} u^{R^{\mathrm{in}}}_{ck'k} v^{X^{R_{\mathrm{in}}}}_{cek'k} \qquad \forall\; e\; k \in K_R \tag{9.38}$$

$$M_{ek} = M^{\mathrm{total}}_{k} x_{eks} \qquad \forall e, k \in K_R \tag{9.39}$$

$$\sum_{e} x_{ek} = 1 \qquad \forall k \in K_R \tag{9.40}$$

$$T_k = u^{T}_{k} \qquad \forall k \in K_R \tag{9.41}$$

$$V^{\mathrm{mol}}_{k} = f^{\mathrm{Vol}}(\mathbf{x}_k, T_k, P_k) \qquad \forall\; k \in K_R \tag{9.42}$$

$$V^{\mathrm{mol}}_{k} M^{\mathrm{total}}_{k} = V_k \qquad \forall k \in K_R \tag{9.43}$$

$$C_{ek} V_k = M_{ek} \qquad \forall e, k \in K_R \tag{9.44}$$

$$\mathrm{rate}_{kr} = C_{A,k} \kappa_r e^{\frac{-E_{A_r}}{RT}} \qquad \forall\; k \in K_R,\; r = 1,2 \tag{9.45}$$

$$\mathrm{rate}_{kr} = C_{I,k} \kappa_r e^{\frac{-E_{A_r}}{RT}} \qquad \forall\; k \in K_R,\; r = 3,4 \tag{9.46}$$

Inequality Path Constraints:

$$310 \leq u^{T}_{k} \leq 450 \qquad \forall k \in K_R \tag{9.47}$$

$$y^{Rxn}_{k} V_k \leq \sum_{i}\sum_{n} y^{R}_{ikn} n \hat{V}^{\mathrm{Vol}}_{i} \qquad \forall k \in K_R \tag{9.48}$$

$$2M_{A,k} \leq M_{B,k} \qquad \forall\; k \in K_R \tag{9.49}$$

Initial Time Constraints:

$$M_{ek} = \sum_{c}\sum_{k'} v^{R_{\mathrm{init}}}_{ck'k} v^{X^{R_{\mathrm{in}}}}_{cek'k} + v^{S_{\mathrm{init}}}_{ek} \qquad \forall\; e, k \in K_R \tag{9.50}$$

$$M^{R^{\mathrm{in}}}_{ck'k} = v^{R_{\mathrm{init}}}_{ck'k} \qquad \forall\; c, k \in K_R, k' \in K_D \tag{9.51}$$

$$M^{S^{\mathrm{in}}}_{ek} = v^{S_{\mathrm{init}}}_{ek} \qquad \forall\; e, k \in K_R \tag{9.52}$$

Final Time Constraints:

$$\sum_{e} M_{ek} = v^{M^{R_{\mathrm{out}}}}_{k} \qquad \forall\; k \in K_R \tag{9.53}$$

$$M_{ek} = v^{M^{R_{\mathrm{out}}}}_{k} v^{X^{R_{\mathrm{out}}}}_{ek} \qquad \forall\; e, k \in K_R \tag{9.54}$$

$$v_{ck'k}^{M^{R_{\text{in}}}} = M_{ck'k}^{R^{\text{in}}} \qquad \forall\ c, k \in K_R, k' \in K_D \tag{9.55}$$

$$v_{ek}^{S} = M_{ek}^{S^{\text{in}}} \qquad \forall\ e, k \in K_R \tag{9.56}$$

The design constraint on the solvent to reactant ratio can be expressed using the following point constraint for each of the reaction tasks.

$$15 \left(\sum_{c} \sum_{e \in \{A,I\}} \sum_{k' \in K_D} v_{ck'k}^{R_{\text{init}}} v_{cek'k}^{X^{R_{\text{in}}}} + \sum_{e \in \{A\}} v_{ek}^{S_{\text{init}}} \right) \leq \left(\sum_{c} \sum_{e \in \{B,W_1,W_2\}} \sum_{k' \in K_D} v_{ck'k}^{R_{\text{init}}} v_{cek'k}^{X^{R_{\text{in}}}} + \sum_{e \in \{B\}} v_{ek}^{S_{\text{init}}} \right) \qquad \forall\ k \in K_R \tag{9.57}$$

We employ constraints expressed in terms of the integer time invariant parameters to define the process structure and a feasible allocation of equipment.

<u>Point Constraints:</u>

$$\sum_{i \in I_D} \sum_{n}^{N_i} y_{ikn}^{C} = y_k^D \qquad \forall k \in K_D \tag{9.58}$$

$$\sum_{i \in I_R} \sum_{n}^{N_i} z_{ikn}^{R} = y_k^{Rxn} \qquad \forall k \in K_D \tag{9.59}$$

$$V_k^{\text{vapor}} = \sum_{i \in I_D} \sum_{n}^{N_i} y_{ikn}^{C} n \hat{V}_i^{\text{vapor}} \qquad \forall k \in K_D \tag{9.60}$$

$$v^{t_{\text{cycle}}} \geq t_k^f + y_k^D \left(t^{\text{fill}} + t^{\text{empty}} + t^{\text{reflux}} \right) \qquad \forall k \in K_D \tag{9.61}$$

$$v_k^{t^{\text{merged}}} = t_k^f + y_{k-1}^M v_{k-2}^{t^{\text{merged}}} \qquad \forall k \in K_R \tag{9.62}$$

$$v^{t^{\text{cycle}}} \geq v_k^{t^{\text{merged}}} + t^{\text{fill}} + t^{\text{empty}} \qquad \forall\ k \in K_R \tag{9.63}$$

$$y_k^D \leq \sum_{i \in I_R} \sum_{n=1}^{N_i} z_{ikn}^{R} \qquad \forall\ k \in K_R \tag{9.64}$$

$$z_{ikn}^{R} = z_{i,k-2,n}^{R} y_{k-1}^{M} + y_{ikn}^{R} \qquad \forall\ i \in I_R, k \in K_R \tag{9.65}$$

$$y_k^M \leq y_k^D \qquad \forall\ k \in K_R \tag{9.66}$$

$$1 \geq y^M_{k-1} + \sum_n^{N_i} y^R_{ikn} \qquad \forall\, i \in I_R, k > 1 \in K_R \tag{9.67}$$

Note that (9.62) assumes that the tasks are ordered R1, D1, R2, D2, etc. In this example the set K contains only two reaction and distillation tasks, so the subscript $k-2$ refers for $k \in K_R$ refers to the previous reaction task and $k-1$ refers to the previous distillation. Mass balances are enforced around each of the tanks used to represent the material that is transferred from one task to another, and (9.69) ensures that a fraction of all recycled material is purged.

$$v^{M^{\text{cut}}}_{ck} = \sum_{k' \in K_R} v^{M^{R_{\text{in}}}}_{ckk'} + \sum_{k' \in K_D} v^{CM}_{ckk'} + v^{CP}_{ck} + v^{CW}_{ck} \qquad \forall\, c,\ k \in K_D \tag{9.68}$$

$$v^{CP}_{ck} + v^{CW}_{ck} \geq X^{\text{purge}} \left(\sum_{k' \in K_R : k' < k} v^{M^{R_{\text{in}}}}_{ckk'} + \sum_{k' \in K_D : k' \leq k} v^{CM}_{ckk'} \right) \qquad \forall\, c,\ k \in K_D \tag{9.69}$$

$$v^{X^{R_{\text{in}}}}_{cekk'} = v^{X^{\text{cut}}}_{cek} \qquad \forall\, c,\ e,\ k \in K_D, k' \in K_R \tag{9.70}$$

$$\begin{aligned} v^{M^{\text{mix}}}_{k} &= \sum_{e \in E_R} v^S_{ek} + \sum_e v^{M^{R_{\text{out}}}}_{k-1} \\ &\quad + \sum_c v^{M^{\text{off}}}_{ck} + \sum_c \sum_{k' \in K_D} v^{CM}_{ck'k} \qquad \forall\, k \in K_D \end{aligned} \tag{9.71}$$

$$\begin{aligned} v^{M^{\text{mix}}}_{k} v^{X^{\text{mix}}}_{ek} &= v^S_{ek} + v^{M^{R_{\text{out}}}}_{k-1} v^{X^{R_{\text{out}}}}_{k-1,e} \\ &\quad + \sum_c v^{M^{\text{off}}}_{ck} v^{X^{\text{off}}}_{cek'} + \sum_c \sum_{k' \in K_D} v^{CM}_{ck'k} v^{X^{\text{cut}}}_{cek'} \qquad \forall\, e,\ k \in K_D \end{aligned} \tag{9.72}$$

$$v^S_{ek} = 0 \qquad \forall\, e \notin E_R, k \tag{9.73}$$

The required product purity is enforced using the following constraint:

$$X^{\text{product}} \sum_{k \in K_D} \sum_c \sum_e v^{M^{\text{cut}}}_{ck} v^{X^{\text{cut}}}_{cek} \text{MW}_e \leq \sum_{k \in K_D} \sum_c \sum_{e \in \{P\}} v^{M^{\text{cut}}}_{ck} v^{X^{\text{cut}}}_{cek} \text{MW}_e \tag{9.74}$$

The itemized production costs can be calculated and assigned to time invariant parameters.

$$v^{\text{Raw}} = \sum_{k} \sum_{e \in E_R} v^{S}_{ek} C^{\text{raw}}_{e} \text{MW}_e \tag{9.75}$$

$$v^{\text{Waste}} = \sum_{k \in K_D} \sum_{c} \sum_{e} v^{CW}_{ck} v^{X^{\text{cut}}}_{cek} C^{\text{waste}}_{e} \text{MW}_e \tag{9.76}$$

$$v^{\text{Equip}} = v^{t_{\text{cycle}}} \sum_{k} \left(\sum_{i \in I_R} \sum_{n}^{N_i} y^{R}_{ikn} n C^{\text{Equip}}_{i} + \sum_{i \in I_D} \sum_{n}^{N_i} y^{C}_{ikn} n \right) \tag{9.77}$$

The constraints defined by (9.11–9.77) must be satisfied. The objective is listed below; it defines the manufacturing cost per unit mass of final product.

$$\phi = \frac{v^{\text{Raw}} + v^{\text{Waste}} + v^{\text{Equip}}}{\sum_{k \in K_D} \sum_{c} \sum_{e} v^{CP}_{ck} v^{X^{\text{cut}}}_{cek} \text{MW}_e} \tag{9.78}$$

9.6 Application of the MIDO Decomposition Algorithm

The Master Problem for this example is the MILP screening model for the batch process development problem presented in chapter 4. The screening model determines a feasible equipment allocation and provides initial guesses of the amounts of material held in each of the tanks described by the time invariant parameters of the dynamic optimization.

The Primal Problem defines a dynamic optimization that considers the reaction and distillation tasks and their recycles simultaneously. This dynamic optimization problem is obtained by simply fixing the integer parameters in the above formulation at the value determined by the solution of the corresponding Master Problem. The time invariant parameters and controls are selected to minimize the production cost per unit mass of product.

We highlight several features of this problem that impact the use of dynamic optimization and the application of the proposed iteration technique. First, the structure of the process considered during dynamic optimization can change depending on the values of the discrete variables chosen by the Master Problem. We can simplify the process structure before solving the dynamic optimization subproblem based on the solution of the Master problem. For instance, some of the processing tasks may not exist within the current process structure. In addition, we may want to consider reducing the number of distillation cuts permitted within the current process structure based on the solution of the screening model. Note that the superstructure defined here permits for 5 overhead cuts, but neither solution of the screening model given in chapter 4 required so many. The qualitative behavior of the distillation may change with small changes in the optimization parameters; for instance, a small change in the feed composition to a distillation task may move the feed into a new batch distillation region. For this reason, we treat the active batch distillation as one of the discrete variables characterizing the process structure; this allows us to address some of the behavior known to lead to multimodality in the dynamic optimization during the solution of the Master Problem.

We note two applications for the Primal Bounding Model incorporated within our algorithm. First, the Primal Bounding Model provides us with a formal strategy for employing the solutions from the global optimization of the nonconvex Primal subproblems whenever such techniques are developed.[2] Second, the Primal Bounding Model provides a convenient and efficient way in which to employ a screening model that is posed as a convex MINLP. The solution of a convex MINLP screening model using traditional decomposition approaches such as GBD (Geoffrion, 1972) or OA (Duran and Grossmann, 1986) results in an iteration strategy like the one shown in figure 9-4. Rather than solving the MINLP screening model to completion on each iteration, we can simply employ the NLP used as the Primal subproblem in the decomposition of the MINLP screening model as as the Primal Bounding model. Using this strategy, shown in figure 9-5, the Master problem of the MIDO decomposition is the same as the Relaxed Master problem used to solve the convex MINLP screening model, the Primal Bounding Model is the convex NLP corresponding to the Primal

[2] It should be noted again that the global solutions to the Primal problem may *not* be used to construct valid support functions.

problem used to solve the MINLP screening model, and the Primal problem is the dynamic optimization corresponding to the values of the y determined by the solution of the Master problem. At each iteration, the Master problem is augmented with the support functions obtained from the solution of the Primal Bounding model; we can continue to add constraints to the Master problem until the solution of the Master problem rises above that of the Primal Bounding problem (e.g., the solution of the screening model has been determined). Using this strategy, the MINLP screening model does not need to be solved to convergence at every iteration.

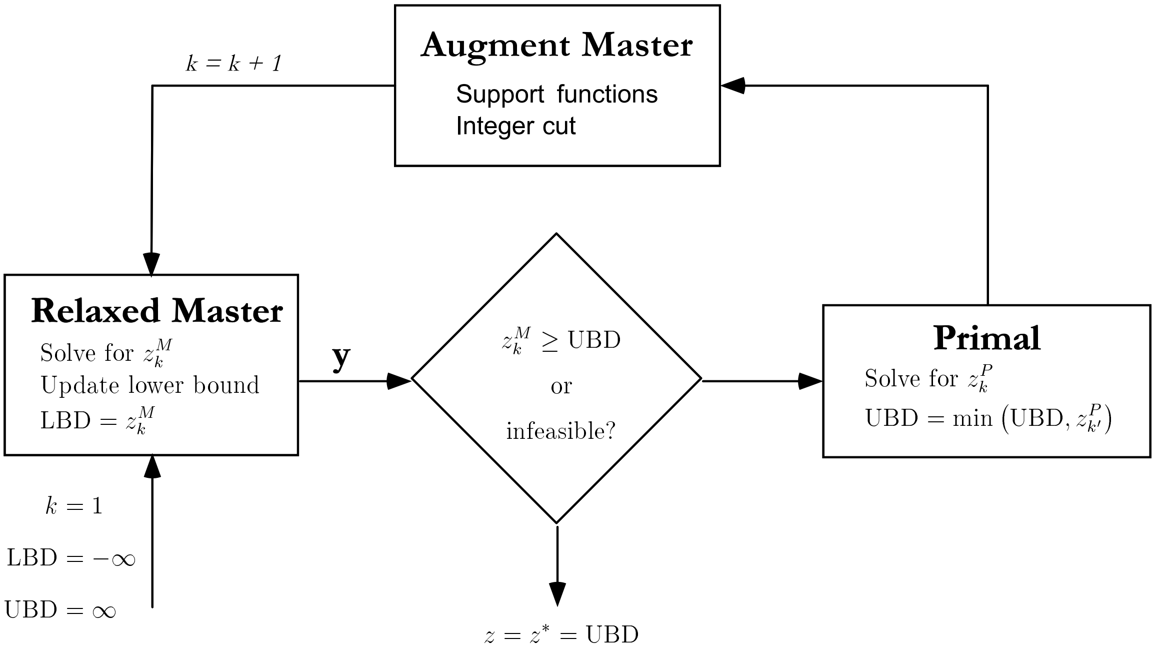

Figure 9-4: Decomposition algorithm employed for MINLP problems.

9.7 Summary

This chapter has defined the class of problems termed *mixed time invariant integer dynamic optimization* problems. We demonstrated that simple extensions of traditional MINLP algorithms are doomed to failure on this class of problems. Instead, we have developed a decomposition algorithm for this class of problems that generalizes the decomposition algorithm employed for batch process development to a

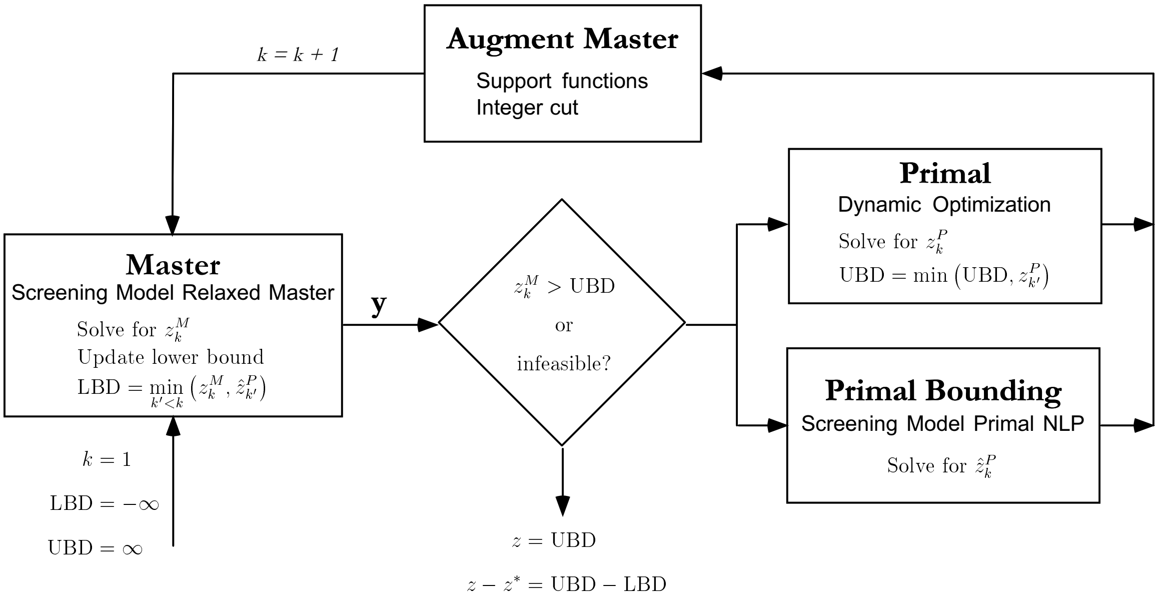

Figure 9-5: MIDO decomposition algorithm when a convex MINLP screening model is employed.

broader class of problems. The algorithm defines a rigorous iteration procedure that guarantees improvement of the solution and potentially avoids explicit enumeration of the entire discrete decision space. The key to the application of this algorithm is the ability to derive a screening model for the particular problem in which we are interested.

We have demonstrated that the batch process development can be formulated as a MIDO problem, and derived the MIDO formulation for the batch process development example of chapter 4.

9.8 Notation

9.8.1 Indexed Sets

C The set of distillation cuts, $c = 1, \ldots NC$

E The set of pure components in the system, $e = 1, \ldots NC$

E_R The set of pure components regarded as raw material that may be supplied

I The set of equipment available in the manufacturing facility

I_D Equipment suitable for distillation tasks $I_D \subseteq I$

I_R Set of equipment suitable for reaction tasks $I_R \subseteq I$

K The set of processing tasks in the superstructure

K_D The set of distillation tasks in the superstructure $K_D \subset K$

K_R The set of reaction tasks in the superstructure $K_R \subset K$

R_k set of reactions occurring in the reaction task in processing train k. In this example, $r = 1, \ldots 4$

S The set of defining the equilibrium stages in the column $s = 1 \ldots NS$

9.8.2 Variables

C_{ek} concentration of component e in the reaction task k

D_{ek} molar flow of component e in the distillate of k

M_{ek} molar holdup of component e in task k

M_{cek}^{cut} molar holdup of component e accumulated in cut c of task k

M_{cek}^{off} molar holdup of component e accumulated in the off cut following cut c of task k

M_k^{total} total molar holdup in task k

$M_{ck'k}^{R^{\text{in}}}$ feed to reactor k from cut c in of distillation k'

$M_{ek}^{S^{\text{in}}}$ supply to reactor k of raw material e

rate_{kr} rate of reaction r in task k

V_k total volume of material in task k

V_k^{mol} molar volume of material in task k

V_k^{vapor} effective vapor rate of the columns assigned to distillation task k

x_{ek}, x_{eks} mole fraction of component e in the reactor k, mole fraction of component e in the liquid phase of stage s in distillation k

x_{ek}^D mole fraction of component e in the distillate of task k

y_e mole fraction of component e in the vapor phase

z^R_{kin} n reactors of type i are assigned to reaction task k

9.8.3 Time Invariant Integer Optimization Parameters

y^C_{ikn} n columns of type i are assigned to distillation task k

y^R_{kin} n reactors of type i begin processing reaction task k. If tasks are merged, they may also process reaction task $k + 2$.

y^{Rxn}_k indicates whether reaction task k is performed

y^D_k indicates whether distillation k is performed

y^M_k indicates whether reaction task k is merged with reaction task $k + 2$

9.8.4 Control Variables

$u^{S_{\text{cut}}}_{ck}$ fraction of the distillate of distillation k that is sent to the accumulator for cut c

$u^{R^{\text{in}}}_{ck'k}$ flow rate into reaction task k from the feed tank fed by cut c of distillation task k'

$u^{S^{\text{in}}}_{ek}$ flow rate into reaction task k from the supply tank containing raw material e

$u^{S_{\text{off}}}_{ck}$ fraction of the distillate of distillation k that is sent to the accumulator for off cut c

u^R_k reflux ratio of distillation task

u^T_k temperature profile for reaction task k

9.8.5 Time Invariant Continuous Optimization Parameters

t^f_k the final time (e.g., length of operation) of task k

$v^{CM}_{ckk'}$ the total molar flow from cut c of distillation k to the mixer tank that feeds distillation k'

v^{CP}_{ck} the total molar flow from cut c of distillation k in the product stream

v_{ck}^{CW} the total molar flow from cut c of distillation k in the waste stream

v^{Equip} cost per batch of equipment

$v_{ck}^{M^{\text{cut}}}$ the total accumulation in cut c of distillation k

$v_{cek}^{X^{\text{cut}}}$ the mole fraction of component e in cut c of distillation k

$v_{ck}^{M^{\text{off}}}$ the total accumulation in off cut c of distillation k

$v_{k}^{M^{\text{mix}}}$ the total molar charge to the mixer tank feeding distillation k

$v_{ck'k}^{M^{R_{\text{in}}}}$ the total molar charge to the feed tank of reaction task k taken from cut c of distillation task k'

$v_{k}^{M^{R_{\text{out}}}}$ the total molar holdup in the tank containing the effluent of reaction task k

v_{ek}^{S} the total molar charge raw material e fed to reaction task k

v^{Raw} cost per batch of raw material

$v_{ck'k}^{R_{\text{init}}}$ the initial charge to reaction task k taken from the feed tank fed by cut c of distillation k'

$v_{ek}^{S_{\text{init}}}$ the initial charge to reaction task k from raw material supply tank e

$v^{t^{\text{cycle}}}$ the cycle time for process operating at cyclic steady state and employing no intermediate storage

$v_{k}^{t^{\text{merged}}}$ the processing time for the potentially merged set of reaction tasks ending with reaction task k

v^{Raw} cost per batch of waste disposal

$v_{ek}^{X^{\text{mix}}}$ the mole fraction of component e in mixer feeding distillation k

$v_{cek}^{X^{\text{off}}}$ the mole fraction of component e in off cut c of distillation k

$v_{cek'k}^{X^{R_{\text{in}}}}$ the mole fraction of component e in feed tank of reaction task k fed by cut c of distillation k'

$v_{ek}^{X^{R_{\text{out}}}}$ the mole fraction of component e in tank containing the effluent of reaction task k

9.8.6 Parameters

MW_e molecular weight component e

N_i number of equipment units i in the manufacturing facility

t^{fill} time required to charge one batch of material to an equipment unit

t^{empty} time required to empty one batch of material from an equipment unit

t^{reflux} time required to bring a column to total reflux

$\hat{V}_i$ processing volume of equipment unit i

$\hat{V}_i^{\text{vapor}}$ maximum vapor rate for distillation column $i \in I_D$

ν_{ekr} the stoichiometric coefficient for component e in reaction r of task k

Chapter 10

Conclusions and Recommendations

This thesis has argued the importance of batch process development to the specialty chemical and synthetic pharmaceutical industries. To be effective, manufacturers must rapidly develop efficient processes. The goals of rapid and efficient development are not necessarily mutually exclusive, and we feel that the application of state of the art process modeling technology can reap benefits on these types of problems in spite of the fact that little has been published demonstrating the benefits that can be achieved through the application of detailed process models. This thesis has addressed two of the hurdles to routine application of process modeling technology to batch process development. First, a systematic and rigorous design procedure has been derived that starts with the information provided by the laboratory synthesis of a new product and employs models of the process at two levels of detail. Second, the numerical solution procedures required for the subproblems inside our framework have been improved, making them more robust and efficient.

10.1 Screening Models for Batch Process Development

This thesis has demonstrated that detailed discrete/continuous dynamic models can be employed within a systematic methodology for the design of batch processes. For an effective yet practical approach, the detailed dynamic models should be used in conjunction with simpler models capable of yielding rigorous lower bounds on the cost of the resulting process that serve as design targets. These bounding models have been termed *screening models* since their solution can be used to prune or screen discrete design alternatives that cannot lead to an optimal solution. These models address the discrete design decisions directly and replace the detailed dynamic models with algebraic constraints that bound the dynamic performance. The derivation of these models was discussed in detail in chapter 3 and the models were demonstrated on process development examples in chapters 4 and 5.

Since screening models provide rigorous bounds on the manufacturing cost, they also permit the derivation of the first rigorous iteration strategy for the improvement of the design as discussed in section 2.4. The iteration strategy involves the solution of the screening model followed by the dynamic optimization of the discrete/continuous dynamic process model for the fixed values of the discrete decisions defined by the solution of the screening model. The solution of the screening model provides a target for the dynamic optimization, and the solution of the dynamic optimization (if feasible) defines a process design alternative in detail. An integer cut is then added to the screening model and the procedure is repeated. Since the objective value of every solution to the dynamic optimization must be greater than that of the corresponding screening model, the iteration can be terminated once the current screening model solution increases above the minimum of the previous solutions to the dynamic optimization. It is important to note that this algorithm generalizes to a class of mixed-integer dynamic optimization problems, provided that appropriate screening models can be constructed. In chapter 9, we also showed that this approach provides the first rigorous method to address mixed-integer dynamic optimization problems using control vector parameterization for the variational subproblem.

Screening models also have the ability to address some of the synthesis decisions involved in process design. The ability to determine the best processing structure was demonstrated by the case studies. Both case studies demonstrate that the screening models can quickly identify potentially favorable processing structures, so that detailed design efforts can be focused on the most promising alternatives first. In fact, a large number of potential configurations may be eliminated through the use of the screening models. The screening models should capture the dominant operating tradeoffs and design constraints in order to provide useful information, but they need not embed all the tradeoffs in the problem in order to provide useful information. These models provide a convenient framework in which to apply limited information about the process and automatically generate potentially beneficial process alternatives that may not have been intuitively obvious to the engineer. More importantly, they provide a good starting point for detailed design and a metric against which existing designs can be measured. In fact, the information provided by the screening model may be all that is required to demonstrate that further design efforts are unwarranted.

10.2 Numerical Issues in the Simulation and Optimization of Hybrid Discrete/Continuous Dynamic Systems

State-of-the-art process modeling environments place extremely high expectations on the efficiency and robustness of the numerical solution procedures. This research has improved both the robustness and efficiency of the integration techniques employed during the simulation and optimization of hybrid discrete/continuous dynamic systems. These improvements have been incorporated within DSL48S, a version of DASSL (Petzold, 1982a) for large sparse unstructured systems of DAEs, which was developed as part of this research. The code employs the MA48 linear algebra routines, works with a combined analytic and numerical Jacobian matrix, and has incorporated the automated scaling and efficient initialization techniques described within this thesis. The code also contains an efficient method for sensitivity analysis that was developed by Feehery and Barton (1997). The code has been implemented within ABACUSS and is substantially more robust and efficient than the version of DASOLV

(Jarvis and Pantelides, 1992) employed within ABACUSS on the wide range of simulations on which it has been tested.

Simulations of the batch distillation of wide-boiling azeotropic mixtures uncovered some of these numerical difficulties. Our investigation determined that the observed problems were caused by an ill-conditioned corrector iteration matrix. We found that scaling of these models, even simply changing the units of the model variables, helped the integration procedures substantially. This motivated us to pursue automatic scaling techniques to improve the solution procedure. We demonstrated that the desired variable scaling is dictated by the user defined error tolerances, and proved that the implemented row scaling brings the two norm condition number of the scaled iteration matrix to within a factor of $2\sqrt{q}$ of the minimum possible, where q represents the maximum number of nonzero entries in any column of the matrix. The automated scaling techniques make the model better scaled than any user implemented scaling resulting from the selection of appropriate units of measurement. In addition, the scaling can automatically adapt to variable values changing over many orders of magnitude during a simulation — a typical occurrence during the simulation of batch processes. The scaling also permits the code to automatically determine whether the potential exists for the truncation error control within the integration method to break down.

Hybrid discrete/continuous dynamic simulation and dynamic optimization requires the integration code embedded within a simulation environment (e.g., ABACUSS) to be started many times during a particular simulation or optimization calculation. On these types of problems, the performance of the integration code during the initial phase of the integration becomes more important. Chapter 8 describes the initialization procedure developed within this thesis. Before the first integration step is taken, the derivatives of the algebraic variables are determined along with the second derivatives of the differential variables. This enables us to generate a reasonable approximation of the step size that approaches the desired truncation error tolerance. We establish criteria that define the optimal initial step length for a first order BDF method, and we demonstrate that the length of a step that satisfies these criteria can be determined by augmenting the system of equations solved during the corrector iteration on the first integration step. The implementation of this procedure does not require any assumptions about the stability of the method at these step lengths, and it permits the same method for the detection of implicit state events to be employed throughout the integration. Application of this method to hybrid discrete/continuous simulation problems has demonstrated that it reduces the number of Jacobian factorizations required, increasing the efficiency of the integration code. Moreover, it reduces the number of truncation error and convergence failures that are observed.

The screening models derived within this research coupled with the advances in the numerical capabilities of the solution procedures provide the engineer facing the batch process development problem with a new and powerful approach, along with the tools required to implement it.

10.3 Recommendations for Future Research

Detailed hybrid discrete/continuous dynamic models of batch processes require an accurate representation of the phenomena of interest. For networks of reaction and distillation tasks, this requires a large amount of data to represent the physical properties and vapor liquid equilibrium for the system components, and to describe the kinetics of the reactions occurring within the process. Since these processes typically involve some chemical species for which little information may be available within existing databases, techniques to gather this data efficiently or to estimate the key properties are required. In addition, the sensitivity of the resulting design to uncertainty in this data must be analyzed. The sensitivity of the resulting design with respect to these parameters can easily be performed within simulation environments such as ABACUSS which can calculate parametric sensitivities. These sensitivities can be used to identify the parameters with the biggest impact on the process design and to direct experimental efforts to reduce the uncertainty in the most influential parameters. The benefits that can be obtained through the use of process models provide a motivation for obtaining this data. Some of the chemical species, such as solvents and familiar reagents may be common to many processes, so their properties may be well understood. However, the reaction kinetics and VLE data are required for the new components. The application of both computational chemistry (Bruneton *et al.*, 1997) and dynamic optimization will enable reaction kinetics to be determined from calorimetric data in a routine fashion. Similarly, advances in calorimeter technology coupled with the increased use of in situ infrared spectroscopy have greatly reduced the experimental effort required to obtain this data. Computational chemistry can be employed to screen potential reaction mechanisms, and dynamic optimization can be employed to perform the parameter estimation. In addition, it may be possible to regress or refine binary VLE parameters from data obtained from laboratory batch distillation columns using dynamic parameter estimation. For modeling to be routinely applied to new processes, efficient methods are required to obtain data. These techniques should be investigated further, now that methods that require the data are available as process design tools.

The screening models that have been developed within this research address processes comprised of only reaction and distillation tasks. However, many specialty chemical and synthetic pharmaceutical processes contain other unit operations, such as crystallization, extraction, filtration, solvent switch, fermentation, and drying. Routine application of detailed dynamic simulation and optimization to batch processes requires development of model libraries for these common batch processing operations. In addition, to increase the applicability of the screening models, suitable models that bound the dynamics of these operations should be developed. Furthermore, to deal with the detailed modeling of operations such as crystallization, improvements to hybrid modeling environments are required to handle the distribution of the particle sizes. More importantly, the separation targets only consider homogeneous mixtures, yet many specialty chemical processes contain heterogeneous (LLE or VLLE) mixtures. To extend the modeling approach to such systems, the batch distillation targeting theory must be extended to heterogeneous systems, and capabilities for discrete/continuous dynamic models to detect the appearance and disappearance of liquid phases and to change the model appropriately during the dynamic simulation are required. Both are active research topics.

A systematic or automated methodology to derive targets for the reaction tasks is also desired. Currently the targets are developed on a case by case basis by identifying key operating tradeoffs and capturing these within the screening models (see chapters 4 and 5). Reactor targeting techniques have been developed to define the attainable region that can be achieved by different continuous reactor configurations (PFRs and CSTRs), but this research relies on geometric arguments and has only been able to consider two and three dimensional (component) systems (Hildebrandt and Glasser, 1990; Hildebrandt *et al.*, 1990; Glasser *et al.*, 1992; Hildebrandt and Biegler, 1995). Other research has examined defining the reactor targets via the solution of a dynamic optimization problem (Hatipoglu and Rippin, 1984; Balakrishna and Biegler, 1992a; Balakrishna and Biegler, 1992b; Balakrishna and Biegler, 1993; Sund and Lien, 1996). However, the inherently nonconvex nature of the dynamic optimization presents a major problem if the desired rigorous bounding properties are to be guaranteed. Extensions of the geometric approaches to handle higher dimensional systems and to account for optimal temperature profiles and feed addition rates for fed batch reactors may permit a systematic methodology to develop targets for the reactors. A key to this development is the extension of current geometric techniques to capture the time/temperature tradeoffs in conjunction with the conversions between competing reactions.

The fact that screening models enable a rigorous iteration procedure for mixed integer dynamic optimization (MIDO) problems opens up the possibility of deriving screening models to address other classes of MIDO problems such as the synthesis of operating procedures (Rivas and Rudd, 1974). The goal in the synthesis of operating procedures is to define a cost effective, safe, and operationally feasible way to move the process from one operating state to another. This requires sequences of valve and pump operations to transfer material between the equipment items allocated to processing tasks through complex piping networks (Foulkes *et al.*, 1988), and the manipulation of control profiles for the processing units. Synthesizing such sequences can be a difficult task where the associated risks are high; products or valuable intermediates may be contaminated by incorrect operations, or extremely dangerous situations might be created from accidentally mixing chemicals that should be kept apart (Lakshmanan and Stephanopoulos, 1990). In batch plants, it requires procedures to ensure the safety of operations such as the startup of batch distillation columns and the charging of reactants given the control system implemented in the plant. Validation alone requires detailed hybrid simulation of the entire process (Crooks and Macchietto, 1992). Simulation provides the means to evaluate the feasibility of proposed procedures, but the synthesis of these procedures defines a dynamic optimization problem in terms of both discrete and continuous decisions. The continuous decisions relate to the set points for controllers, the flows of steam and cooling water (or the settings for the control valves) to different equipment units, and the definition of control profiles for all of the units within the facility. Discrete decisions relate to valves that are either open or closed, pumps and other equipment that is either on or off, control systems that are either in manual or automatic mode, and whether specific units are used or idle. These problems remain very difficult, but the recent results can yield parametric sensitivities for hybrid discrete/continuous dynamic systems (Barton, 1996). Coupling these sensitivities with screening models may enable these problems to be investigated within a MIDO framework.

Batch process development also requires a large quantity of data to describe the equipment within the processing facility, to describe the processing operations, to define the operating policies, and to define the schedule for the manufacturing facilities equipment. The ability to manage this data is a difficult task, and a software environment both to manage the data and access the appropriate numerical tools is required. The environment must have facilities to specify the process models, access the numerical tools, and analyze the results.

A central feature of such an environment is the representation of the batch process and the batch plant. The tools incorporated within the environment will rely upon seamless communication with the information stored in these models. The plant can be represented by a graph where the nodes represent physical items of equipment (including reactors, distillation columns, valves, pumps, sections of pipe, etc.). The arcs of the graph represent the connections between these items. The graph should contain the information typically found in a process and instrumentation diagram. The process can also be represented by a graph; a hierarchical state task network (STN) can be used to describe the process at multiple levels of detail (Allgor *et al.*, 1996). Processing tasks will require models, possibly several models at different levels of detail, to describe the physical and chemical transformations that can occur within the task.

Since both the plant and the process will be described by graphs, it is natural to build these models using graphical methods. Furthermore, high level communication between the engineer and the software is essential for the development environment to speed the engineer's design procedure. Earlier in this research we investigated a framework with which to represent the processing facility. We developed the notion of an equipment class. Each item of equipment within the processing facility is described as an instance of a particular equipment class. The equipment classes can be constructed in a hierarchical fashion. Each equipment class is specifies the class that it inherits from, new attributes for the class, and the instances of contained classes and their interconnections. The plant was described as an instance of an equipment class. We developed a graphical environment within which the equipment classes could be constructed, but this area of the research was not pursued further.

For many specialty chemical and synthetic pharmaceutical processes the management of the data representing the process is a major concern. Creating an environment that provides a framework to capture and store the knowledge about the process and the current process design is essential. The model would then provide a unified framework in which to represent the process over the entire lifetime of the process. This environment can then generate descriptions of the process recipe in the form required for the individuals working on the process, and maintain versions of the process that are constructed as the design procedure continues. Process design tools for batch processes have recently been developed such as the BatchDesign-Kit (Linninger *et al.*, 1996a; Linninger *et al.*, 1996b) and BATCH PLUS$^{\mathrm{TM}}$ (Aspen Technology, Inc., 1997), but they need to evolve to a state that permits the seamless integration of process models at several levels of detail and permits access to both sophisticated scheduling algorithms and detailed dynamic simulation and optimization.

Appendix A

Matrix and Vector Norm Proofs

This chapter contains proves some vector and norm properties that are exploited within the thesis. These theorems are not new, but the proofs have been provided to aid the reader in understanding the proofs contained in the body of the thesis.

Theorem A.1. *For $D \in \mathcal{D}_n$, $x \in X$, and $\|\cdot\|_\alpha$ an* absolute *(Bauer* et al.*, 1961) vector norm, then the following holds:*

$$\min_j |d_{jj}| \, \|x\| \leq \|Dx\| \leq \max_j |d_{jj}| \, \|x\| \tag{A.1}$$

Proof. Define the vectors $z = \min_j |d_{jj}| x$ and $\breve{z} = \max_j |d_{jj}| x$. For every component i of Dx, the following hold.

$$\begin{aligned} \min_j |d_{jj}||x_i| &\leq (Dx)_i \leq & \max_j |d_{jj}||x_i| \quad & \text{(A.2)} \\ |z_i| &\leq (Dx)_i \leq & |\breve{z}_i| \quad & \text{(A.3)} \end{aligned}$$

Since the norm is absolute, (A.3) shows that $\|x\| \leq \|Dx\| \leq \|\breve{z}\|$. □

Note that all of the Hölder p-norms are absolute, so this holds for the norms in which we are interested.

Theorem A.2. $\max_j \left\|(A^H)_j\right\|_2 \leq \|A\|_2$

Proof. Let e_j represent the jth column of the identity matrix. We also know that $\|A\|_2 = \left\|A^H\right\|_2$ (Bauer *et al.*, 1961).

$$\begin{aligned}
\max_j \left\|(A^H)_j\right\|_2 &= \max_j \left\|A^H e_j\right\|_2 && \text{(A.4)}\\
&= \max_{x \in \{e_j:\ j=1,2,\ldots m\}} \frac{\left\|A^H x\right\|_2}{\|x\|_2} && \text{(A.5)}\\
&\leq \sup_{x \in X} \frac{\left\|A^H x\right\|_2}{\|x\|_2} && \text{(A.6)}\\
&= \left\|A^H\right\|_2 && \text{(A.7)}
\end{aligned}$$

□

Theorem A.3. $\|A\|_2 \leq \sqrt{m} \left\|(A^H)_j\right\|_2$

Proof. We use the relationship between the two norm and the Froebenius norm (Golub and Van Loan, 1989).

$$\begin{aligned}
\|A\|_2 \leq \|A\|_F &= \sqrt{\sum_i \sum_j a_{ij}} && \text{(A.8)}\\
&= \sqrt{\|(A^H)_1\|_2^2 + \|(A^H)_2\|_2^2 + \ldots + \|(A^H)_m\|_2^2} && \text{(A.9)}\\
&\leq \sqrt{m \max_j \|(A^H)_j\|_2^2} && \text{(A.10)}\\
&= \sqrt{m} \max_j \left\|(A^H)_j\right\|_2 && \text{(A.11)}
\end{aligned}$$

□

The following theorem is used to avoid the square root operation when calculating the two norm of the rows of the iteration matrix during the automated scaling algorithm.

Theorem A.4. *Define $\mathbb{Z}$ as the set of integers. Let $p, q \in \mathbb{Z}$ and $x, r, \varepsilon \in \mathbb{R}$ where $0 \leq \varepsilon < 1$ and $q > 0$. We define $p = \lfloor x \rfloor$ and ε, so $x = \lfloor x \rfloor + \varepsilon = p + \varepsilon$. We define $r = p/q - \lfloor p/q \rfloor \leq (q-1)/q$. The following holds for all $q > 0$:*

$$\left\lfloor \frac{x}{q} \right\rfloor = \left\lfloor \frac{\lfloor x \rfloor}{q} \right\rfloor \qquad \text{(A.12)}$$

Proof. Expand the right hand side and use the relationship that for $a \in \mathbb{Z}$ and $y \in \mathbb{R}$, $\lfloor a + y \rfloor = \lfloor a \rfloor + \lfloor y \rfloor$.

$$\begin{aligned}
\left\lfloor \frac{x}{q} \right\rfloor &= \left\lfloor \frac{p+\varepsilon}{q} \right\rfloor && \text{(A.13)}\\
&= \left\lfloor \left\lfloor \frac{p}{q} \right\rfloor + \frac{r+\varepsilon}{q} \right\rfloor && \text{(A.14)}\\
&\leq \left\lfloor \left\lfloor \frac{p}{q} \right\rfloor + \frac{q-1+\varepsilon}{q} \right\rfloor && \text{(A.15)}\\
&= \left\lfloor \frac{p}{q} \right\rfloor + \left\lfloor \frac{q-1+\varepsilon}{q} \right\rfloor && \text{(A.16)}\\
&= \left\lfloor \frac{p}{q} \right\rfloor = \left\lfloor \frac{\lfloor x \rfloor}{q} \right\rfloor && \text{(A.17)}
\end{aligned}$$

However, by definition $\lfloor x/q \rfloor \geq \lfloor \lfloor x \rfloor / q \rfloor$, so equality must hold. □

A.1 Comments on condition numbers, inf, sup, and rectangular matrices

The ratio $\sup_{\alpha\beta}(A)/\inf_{\alpha\beta}(A)$ can be defined for any matrix A. When A is square and invertible, we can define A^{-1}. Let $v = Ax$ and we have $\inf \|Ax\| / \|x\| = \inf \|v\| / \|A^{-1}v\|$. This can be rewritten as $\sup \|A^{-1}v\| / \|v\| = \|A^{-1}\|$. For the two norm this quantity takes on the familiar form $\|A\|_2 \|A^{-1}\|_2$, and we can see that the condition number is the ratio of the largest to smallest singular values. We define $\|A^{-1}\|_2 = \infty$ for singular A, which makes sense when we examine the the SVD of A. The same definition is employed for other norms, knowing that consistent norms are related (Golub and Van Loan, 1989). The condition number defined on the two norm has useful properties since it relates to the diagonal form of the matrix.

Appendix B

Solution of an augmented system of linear equations

Consider the following linear system of equations:

$$\begin{bmatrix} A & c \\ r^T & e \end{bmatrix} \begin{bmatrix} z \\ h \end{bmatrix} = \begin{bmatrix} b \\ f \end{bmatrix} \tag{B.1}$$

where $A \in \mathbb{R}^{n \times n}$, $c, r, z, b \in \mathbb{R}^n$, and $f, h, e \in \mathbb{R}^1$. Assume that A is nonsingular and that it has been factored, so that linear systems of the form $Az = b$ can be solved efficiently. This implies that (B.1) can be solved without factoring the entire matrix. The first n rows of (B.1) can be expressed as follows:

$$z = A^{-1}(b - hc) \tag{B.2}$$

Substituting the expression for x from (B.2) into the last row of (B.1) yields the following:

$$h(e - r^T A^{-1} c) + r^T A^{-1} b = f \tag{B.3}$$

which can be rearranged to solve for y:

$$h = \frac{r^T A^{-1} b - f}{r^T A^{-1} c - e} \tag{B.4}$$

Thus, the solution (B.1) is given by (B.2) and (B.4). Calculating the answer requires the forward and back substitution of two factored linear systems ($Av_1 = b$ and $Av_2 = c$), two dot products, and one saxpy call (Golub and Van Loan, 1989).

Appendix C

Time derivatives of the algebraic sensitivity variables

The linear system that defines the time derivatives of the algebraic sensitivity variables differs from (8.7) in only the right hand side vector.

The sensitivity equations corresponding to the parameter p are defined by taking the differential of the DAE model with respect to p as follows (since $\partial u/\partial p = 0$):

$$\frac{\partial f}{\partial \dot{x}}\frac{\partial \dot{x}}{\partial p} + \frac{\partial f}{\partial x}\frac{\partial x}{\partial p} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial p} + \frac{\partial f}{\partial p} = 0 \tag{C.1}$$

The values $\partial x/\partial p$ and $\partial y/\partial p$ are referred to as the differential and algebraic sensitivity variables for the parameter p. Differentiating (C.1) with respect to time yields the following:

$$\frac{\partial^2 f}{\partial \dot{x}\partial t}\frac{\partial \dot{x}}{\partial p} + \frac{\partial f}{\partial \dot{x}}\frac{\partial \ddot{x}}{\partial p} + \frac{\partial^2 f}{\partial x\partial t}\frac{\partial x}{\partial p} + \frac{\partial f}{\partial x}\frac{\partial \dot{x}}{\partial p} + \frac{\partial^2 f}{\partial y\partial t}\frac{\partial y}{\partial p} + \frac{\partial f}{\partial y}\frac{\partial \dot{y}}{\partial p} + \frac{\partial^2 f}{\partial p\partial t} \tag{C.2}$$

which can be evaluated at t_o to define the derivatives of the algebraic sensitivity variables as shown below:

$$\begin{aligned}
\begin{bmatrix} \frac{\partial f}{\partial \dot{x}} & \frac{\partial f}{\partial y} \end{bmatrix}_{t=t_o} \begin{bmatrix} \frac{\partial \ddot{x}}{\partial p} \\ \frac{\partial \dot{y}}{\partial p} \end{bmatrix}_{t=t_o} &= -\left(\frac{\partial^2 f}{\partial \dot{x} \partial t} + \frac{\partial f}{\partial x} \right)_{t=t_o} \left. \frac{\partial \dot{x}}{\partial p} \right|_{t=t_o} \\
&\quad - \left. \frac{\partial^2 f}{\partial x \partial t} \right|_{t=t_o} \left. \frac{\partial x}{\partial p} \right|_{t=t_o} - \left. \frac{\partial^2 f}{\partial y \partial t} \right|_{t=t_o} \left. \frac{\partial y}{\partial p} \right|_{t=t_o} - \left. \frac{\partial^2 f}{\partial p \partial t} \right|_{t=t_o} \qquad \text{(C.3)}
\end{aligned}$$

The right hand side of (C.3) needs to be provided to the integration code in order to solve for the derivatives of the algebraic sensitivity variables. Although this requires the second order derivatives of the DAE equations, only equations in which time appears explicitly can contribute to the right hand side (i.e., those containing controls or time). Since time often does not appear explicitly, or may only appear in a few equations, the right hand side vector may be easily calculated in many cases. If time appears explicitly in many equations, the effort required to evaluate the right hand side may not be worth the benefits that are obtained by calculating the derivatives of the algebraic sensitivity variables.

If a routine is provided to calculate the right hand side of (C.3), DSL48S will determine the derivatives of the algebraic sensitivities.

Appendix D

Review of Batch Plant Design Literature

What is commonly referred to as the "Batch Plant Design Problem" is the problem of minimizing the investment cost required to purchase a set of equipment items in order to meet a fixed set of production requirements. This deterministic design problem was first addressed by Ketner (1960) and later by Loonkar and Robinson (1970; 1972). Their original formulations of the batch plant design problem were fairly restrictive with respect to the types of operating alternatives that were considered; they considered only simple processing scenarios. Subsequent research has focused on considering more sophisticated operating strategies during the plant design. The progress on this problem has recently been reviewed by Rippin (1993) and Reklaitis (1989). The growth in the list of publications in this area since Rippin's previous review (1983a) demonstrates that a significant amount of research has been conducted over the last ten years. However, progress in these areas has been incremental, and to this date a rigorous formulation of the problem that accounts for all possible alternatives has not been found.

The objective in the plant design problem is typically stated as the minimization of investment cost which consists of the costs to install the batch, semicontinuous, and continuous equipment in the processing facility; detailed aspects of the design such as the instrumentation, and the connecting pipes and valves are typically ignored. Recently, however, systematic approaches to determine the optimal layout of the processing equipment has been considered (Jayakumar and Reklaitis, 1994; Jayakumar and Reklaitis, 1996). This objective is minimized subject to production and operating constraints. The production constraints usually specify the amount of each product required within a given time horizon, and the operating constraints are used to insure the feasible operation of the proposed design. The engineer is faced with the task of determining a way to minimize the objective within the operating and production constraints imposed on the system. These engineering decisions have been decomposed by Reklaitis (1989) into the following hierarchy of design subproblems:

1. Determine the recipe task network.
2. Select the best operating strategy (single product, multiproduct, or multipurpose).
3. Create the equipment configuration (storage locations, parallel units, and the assignment of units to processing stages).
4. Size equipment and determine the cost of the design.

When analyzing the progress that has been achieved on the batch plant design problem, it may be useful to examine the improvements made to each of the subproblems defined above. This view of the design problem is useful when analyzing the way in which particular instances of the design problem are solved. In fact, many of the heuristic solution methodologies applied to this problem utilize a similar decomposition of the decision tasks.

Rippin (1983a) has proposed a different way to classify batch processing problems that focuses on what facets of the batch process design is to be addressed. The components of the problem statement dictate what decisions and/or assumptions the engineer must make. The components of the problem considered by Rippin are organized into the following classes:

product requirements Requirements for products are specified in quantity and time. Time horizons for products may be given, or the production may have to meet a set of delivery dates. In some cases the demand may be uncertain requiring added flexibility in the design.

process tasks These are identifiable operations carried out on the system components. The mode of execution (i.e., batch) must be given along with the sequence of tasks to be carried out. The performance of each task (i.e., operating conditions) must also be determined. Often times these items are given as a product "recipe" and assumed fixed during the design procedure.

system structure The types, sizes, and number of equipment items must be determined. Tasks must be assigned to equipment items or groups. Constraints on the sequencing of tasks must be enforced. Product campaigns or a detailed production plan is required. The role of intermediate storage must also be considered.

design objective The cost objective must be formulated.

solution methods Exact and heuristic methods are considered. The tradeoffs between efficiency and accuracy must be balanced.

Rippin partitions each of these categories into its essential elements, and examines the way in which each of the elements is fulfilled. The variation in the way these elements are fulfilled describes the different design problems that have been addressed. Rippin's method of classification is useful for analyzing which design problems have been studied; it demonstrates that improvements to the original "batch design" problem have focused primarily on the following areas:

- Considering multiple products.
- Including the cost of semicontinuous units.
- Using parallel units both in-phase and out-of-phase.
- Utilizing multipurpose operating procedures.
- Varying the task to stage assignment.
- Accounting for discrete equipment sizes.

- Incorporating intermediate storage and accounting for its cost.

- Planning for uncertainty in the design.

- Varying the solution methods used.

The impact each of these items has had on the formulation and solution of the "batch plant design problem" will be covered in the rest of this section.

D.1 Multiple Products

The first attempt at systematically minimizing the capital investment for a batch process was undertaken by Loonkar and Robinson (1970). They considered the capital cost of the batch and semicontinuous equipment for a single product plant. Given a "recipe" for the product consisting of a sequence of operations and processing times, the challenge was to find the equipment sizes which minimize the cost of the plant. The optimal cost was determined using calculus to derive the first order necessary conditions for optimality; the resulting set of equations was then solved to obtain an optimum of the convex programming problem. The authors extended this idea to multiproduct plants producing products in single product campaigns (Robinson and Loonkar, 1972). All the products manufactured must follow the same sequence of processing steps, but some of the products may not require certain processing stages. A multivariable direct search was used to solve for the optimum value of the objective function. In both of these formulations it was assumed that the equipment is available in a continuous range of sizes.

With few exceptions, the batch process design problems addressed after 1972 have dealt exclusively with multiproduct and multipurpose plants. One notable exception is the work of Yeh and Reklaitis (1987). In their work the authors focused on issues relating to the synthesis of the process structure, so they chose a single product plant to eliminate many of the production planning aspects which complicate the problem.

D.2 Semicontinuous units

The initial work on the batch process design problem considered the cost of semicontinuous equipment in the objective function (Ketner, 1960; Loonkar and Robinson, 1970; Robinson and Loonkar, 1972). Most of the recent work addressing this problem has ignored the costing and sizing of the semicontinuous units based on the following three assumptions:

1. The cost of semicontinuous equipment is negligible compared to the cost of the batch units (Suhami and Mah, 1982).
2. The cost of the semicontinuous equipment is relatively constant over the range of plant configurations being considered in the design problem; therefore, it can be removed from the objective function (Vaselenak *et al.*, 1987).
3. The semicontinuous units will usually be selected based on criteria other than cost such as safety or prior experience (Sparrow *et al.*, 1975).

The cost of batch equipment is typically a function of equipment type and volume and the cost of semicontinuous equipment is expressed as a function of processing rate. In the work of Loonkar and Robinson (1970; 1972) the inclusion of semicontinuous units was necessary to facilitate their solution procedure. Since the operation times of the semicontinuous equipment items are a function of the processing rate, they treated the processing rates of the semicontinuous units as the decision variables of the problem.

In contrast to this, Sparrow *et al.* (1975) chose to ignore the costing and sizing of the semicontinuous units. They state that these units will be chosen on the basis of safety or past experience, and their selection will not in general be dictated by cost. Even though the size of the semicontinuous units will determine the time required to transfer material to and from the vessels, most recent formulations have assumed that the emptying and filling times are fixed and are not functions of the volume of material processed. The processing rate of the equipment can be selected to meet the assumed filling and emptying times without significant changes to the cost of the semicontinuous units that need to be purchased.

One exception to the trend of ignoring the semicontinuous units is the work of Knopf *et al.* (1982). They consider the design of a cottage cheese process in which the cost of operating the semicontinuous units (separator-homogenizer-pasteurizer units) is substantial. The operating and investment costs for these units are considered in the objective function, and these costs drive the design decisions.

More recently, Yeh and Reklaitis (1987) have incorporated the cost and sizing of the semicontinuous units in order to calculate the cycle times of stages of the processing. Their research focused on the synthesis of a superstructure for the batch process, where the selection of the task to stage assignments may have a significant affect on the number of semicontinuous units, such as pumps, which are required. However, they drew no conclusions about the need to accurately express the costing and sizing of the semicontinuous units, although it is clear that the appropriate merging of tasks may eliminate some of the semicontinuous units from the process structure (Patel *et al.*, 1991). Similarly, Modi and Karimi (1989) include semicontinuous units when considering finite intermediate storage. They account for the costing and sizing of the semicontinuous units, but the heuristic procedure which they use to optimize the design is based partly on the assumption that the cost of a semicontinuous unit is far less than that of a batch unit.

While the assumptions regarding the relative costs of the semicontinuous units may be valid under certain circumstances, the treatment of the issue points out that the plant design problem will need to be evaluated for the specific case at hand, and solution procedures should only be applied where they are applicable. The important costs of any design problem will most likely need to be considered on a case to case basis.

D.3 Parallel Units

The processing tasks for a particular product are executed in one or more of the equipment items existing in the plant. Typically, a task or group of tasks is assigned to a processing stage, and then one or more equipment items is assigned to this processing stage for a particular product.

Every processing stage must be assigned an item or group of items in which the processing tasks will be carried out. If one equipment item is assigned to a particular stage, the only design decisions required are the selection of the size and type of unit assigned. However, if more than one unit is assigned in parallel, several decisions must be made.

- How many items in parallel are to be used?
- Are the units to be operated in-phase or out-of-phase?
- Should the units in parallel be of the same size?

Considering parallel units in the process structure increases the combinatorial complexity of the design problem and may require the use of tailored optimization procedures. Since the complexity typically governs the performance of the solution procedures, problem simplifications that reduce the complexity, or tailored solution algorithms are often required. If the type of parallel units considered is limited in some way, then the complexity is reduced and solution methods that take advantage of this simplification may be employed. For this reason, the type of parallel units considered has been limited in much of the research that has been conducted so far.

The first paper to address the use of parallel units was written by Sparrow *et al.* (1975). The authors extended the formulation of Robinson and Loonkar (1972) by considering the addition of identical parallel units at each stage operating out of phase. Units used in this fashion serve to reduce the cycle time of a particular processing stage and have been studied by many other authors (Grossmann and Sargent, 1979; Knopf *et al.*, 1982; Suhami and Mah, 1982; Yeh and Reklaitis, 1987; Vaselenak *et al.*, 1987; Faqir and Karimi, 1989).

The use of parallel units operating in phase has been considered by relatively few studies. Flatz (1980; 1981) mentions the use of in phase parallel units, but his evolutionary design procedure does not systematically attempt to account for their use. Yeh (1987) demonstrates that there is no incentive to use in phase parallel units or units of different size operating in parallel for the design of single product plants.

Papegeorgaki and Reklaitis (1990a; 1990b) are the first to include the use of in phase parallel units in a systematic fashion; their formulation of the multipurpose design problem accounts for the use of in phase parallel units which may differ in size. They formulate the problem as an MINLP with integer variables representing the use of a particular unit for a given task. In phase parallel units are included by forming product groups which can contain units of different sizes. These product groups are then assigned to particular tasks with groups assigned to the same task operating out of phase. However, incorporating this flexibility through the use of integer variables rapidly increases the combinatorial complexity of the problem, rendering current MINLP solution procedures ineffective in the problem solution. In addition to the added complexity, the structure of the constraints yields a problem with a considerable amount of degeneracy which places an additional burden on the solution technique. To avoid some of the degeneracy, additional constraints are added to the formulation. Papageorgaki details an approximate solution procedure designed specifically for this problem (Papageorgaki and Reklaitis, 1990b).

D.4 Multipurpose Plants

The extension of the design problem to multipurpose plants requires fairly detailed aspects of the production planning problem be solved at the design stage, simultaneously with the sizing and structural aspects of the design. The definition of a multipurpose plant has not been agreed upon by all researchers in the field. A multipurpose plant will provide the most flexible plant designs where different batches of the same product may follow different routes through the plant. The following definitions will be used in an attempt to avoid confusion with the use of these terms:

Multiproduct Plant A plant producing more than one product in the form of single product campaigns. Every batch of a given product follows the same route through the plant.

Multipurpose Plant with Single Production Routes The plant consists of many processing stages each having multiple identical batch units each operating in parallel, out of phase. Every product passes through a fixed subset of these stages; thus, each product follows only one production route and every batch of a given product has the same size (Faqir and Karimi, 1989).

Multipurpose Plant with Multiple Production Routes In this case each product may have follow multiple routes through the process. A given product may have multiple batch sizes corresponding to different processing routes (Faqir and Karimi, 1990).

Multi-plant Plant A plant where the equipment items are permanently partitioned into parallel processing trains.

Multipurpose Plant The most general form of processing. A given product need not follow the same processing path from one batch to the next. The same equipment may be reused for different tasks of the same product. Equipment items may be shared between products during the same production campaign.

The initial work attempting to address multipurpose plants was performed by Suhami and Mah (1982). They proposed a design formulation for a multipurpose plant with single production routes; they only considered multiple products within a campaign if they could be processed via parallel production routes. They developed a heuristic procedure to generate sets of "compatible" products, those which do not share any equipment items, to be processed in each campaign. Groups of these compatible product sets were formed such that each group covered all of the products to be produced. For each group of product sets, the appropriate horizon constraints were developed according to some heuristic rules. The resulting MINLP was then solved for each product group. The lowest cost solution was considered the optimum. Several years later, Imai and Nishida (1984) presented a way to systematically determine the best group of product sets. They determined the best group of product sets by solving a set partitioning problem. They generated the horizon time constraints from this set, but did not detail the method used to derive the constraints.

Vaselenak *et al.* (1987) extended the previous work by developing a multiperiod formulation which uses a superstructure to embed all of the possible product configurations. They employ a systematic procedure to derive the maximal set of compatible products which is used to generate the horizon constraints based on the available production time for each equipment unit. These constraints are then "merged" to form an equivalent set of constraints that treats the portion of the horizon time allotted to each product as the independent variables. The resulting superstructure creates a single MINLP which can be solved for the optimum. The authors realize that only in some cases, the "fully merged" case, will the derived horizon constraints result in a relaxed NLP which is convex. Thus, in many of the cases, they cannot guarantee optimality of their solution.

Faqir and Karimi (1989) attack the work of Vaselenak. These authors formulate a programming problem to address the same problem as that of Vaselenak. Their formulation results in a single MINLP, and the formulation does not depend on whether the horizon constraints can be "fully merged" (Vaselenak *et al.*, 1987). This procedure involves fewer variables and constraints than the method of Vaselenak, and the equivalence of their resulting formulation and the original one based on the horizon times of each equipment item can be formally proven. However, the relaxed NLP may not be convex, so this method also cannot guarantee a globally optimal solution. Thus, it can be seen that even for the relatively simple case of single production route multipurpose plants, a globally optimal solution cannot be guaranteed.

Later, Faqir and Karimi (1990) extend this idea to plants with multiple production routes; they consider all of the possible routes that a each product can follow through the plant and group these into sets of compatible production routes. A similar method for deriving the horizon constraints and solving the resulting problem is employed. The resulting problem is more complex, and once again a globally optimal solution cannot be guaranteed. Kiraly *et al.* (1989) address a very similar problem, but they employ a two-step decomposition strategy to arrive at the optimal investment cost. The aforementioned formulations can only handle situations where the equipment items can be allocated to parallel processing trains, each representing a particular production route, in every campaign considered.

D.5 Varying the Task to Stage Assignment

One aspect of batch processes which makes them particularly flexible is the distinction between the plant and the process. A manufacturing process is free utilize the plant in the most appropriate way for every product it produces. While the processing tasks are defined for each product, the way in which to implement them on the plant is not. A processing stage may be assigned single or multiple processing tasks. Yeh and Reklaitis (1987) showed that selecting the optimal task to stage assignment enables more efficient use of the processing equipment, but adds complexity to the design decisions. Even though the optimal assignment may greatly improve the efficiency of a design, the task to stage assignment problem has been overlooked in most of the batch plant design literature.

Papageorgaki and Reklaitis (1990a) are the first to consider the task to stage assignment within the plant design of multipurpose plants. In fact, they are the first to formulate the design for a plant operating in true multipurpose fashion. They account for multiple production routes, the use of an equipment item for different tasks within the same production campaign, and for the use of in phase and out of phase parallel items of equipment at each processing stage. Their formulation seems superior to any others proposed thus far, but the number of integer variables is excessive. They have found that standard MINLP solution techniques fail miserably when applied to this problem, usually yielding solutions which are inferior to those determined by much simpler formulations of the design problem. Since the problem is so difficult to solve, the added flexibility within this formulation cannot be exploited. The authors address this problem by proposing a method to decompose and solve the MINLP to near optimality in a companion paper (Papageorgaki and Reklaitis, 1990b).

D.6 Discrete Equipment Sizes

Much of the equipment purchased for a new batch plant will be chosen from a set of standard equipment sizes. However, in many of the programming formulations of the batch plant design problem, standard equipment sizes have been overlooked in order to facilitate the solution of the problem. In fact, the size of the equipment to be purchased in a batch plant has been considered a continuous variable in many of the formulations of the batch design problem (Robinson and Loonkar, 1972; Grossmann and Sargent, 1979; Suhami and Mah, 1982; Birewar and Grossmann, 1989). These authors usually state that the equipment can be rounded up to the next standard size to produce a realistic feasible design.

On the other hand, some authors have demonstrated that simply rounding the optimal equipment sizes found in a continuous solution up to the next standard size will not necessarily produce an optimal design. In order to account for this, these authors have explicitly considered the discrete equipment sizes in their problem formulations (Sparrow *et al.*, 1975; Flatz, 1980; Faqir and Karimi, 1989). Incorporating the standard equipment sizes explicitly in the problem formulation adds combinatorial complexity to the problem which usually increases the solution time.

Recently, however, the initial justification for incorporating standard equipment sizes, to make the problem more realistic, seems to have given way. Grossmann *et al.* (1992) have shown that many design problems which involve nonlinear separable objective functions and bilinear constraints can be reformulated. The formulations of Voudouris and Grossmann (1992a; 1992b) take advantage of the discrete equipment sizes to transform the NLP (Grossmann and Sargent, 1979) or MINLP (Birewar and Grossmann, 1989) formulations into a mixed integer linear programming problems (MILP). The MILP formulations have the advantage that they can be solved to global optimality. Also, the methods used to solve these problems are more robust than those that attempt to cope with non-convex NLPs.

In many cases the MILP problem contains no more variables than the original problem and can be efficiently solved by taking advantage of the special structure of the design problem. In other cases, such as the formulation of Faqir and Karimi (1989), converting the bilinear constraints into an equivalent set of linear ones requires the addition of many variables. In these situations, the MILP which must be solved is considerably larger than the original MINLP, and the benefits of the transformation are not quite as clear, although global optimality of the solution can still be guaranteed.

D.7 Intermediate Storage

Intermediate storage is often used in batch processes to reduce the amount of idle time processing stages by partitioning the plant into two or more processing trains, each with its own batch size and cycle time. The processing rate of each of the trains must be equal to insure that the size of the intermediate storage remains finite. In order to employ intermediate storage, two basic decisions need to be made:

1. Determine the location of the intermediate storage.
2. Determine the size of intermediate storage required at each location.

In many design problems the cost of intermediate storage is ignored, and the storage is located according to several scenarios — unlimited intermediate storage (UIS), no intermediate storage (NIS), or zero wait (ZW). In such cases the size and cost of the required storage is not of primary concern and is not considered when choosing between design alternatives.

Takamatsu *et al.* (1982) first addressed the problem of determining the amount of storage required to decouple two processing trains. They demonstrated that the minimum required storage is a function of the batch sizes and cycle times of the upstream and downstream units, the initial holdup in the tank, the filling and emptying rates of the storage vessel, and the lag time between the two batch units. They showed that the optimal storage cost is a discontinuous function of the batch sizes and calculated the optimal storage size when the filling and emptying rates of the storage tank are equivalent. The minimum storage size was given as a function of the greatest common measure of the batch sizes and the lag time between units. Karimi and Reklaitis (1983) considered a similar case where the filling and emptying rates of the tank are not equal. They determined the minimum storage requirement by first deriving an analytical solution to the differential equation describing the tank holdup via Fourier series analysis, ignoring any changes in volume upon mixing. With this solution, they determined the minimum and maximum values of the holdup. Karimi and Reklaitis (1985) built upon this work by determining the minimum volume requirement for a processing network in which the number of parallel units both upstream and downstream of the storage may vary. They calculated the lag time policy which minimizes the storage requirement. More importantly, they generated a simple upper bound, independent of the lag time policy employed, for quickly estimating the storage volume requirements.

The optimal sizing of intermediate storage within a multiproduct plant has been addressed by Modi and Karimi (1989). The authors developed a method which considers the cost and sizing of intermediate storage during the design stage of a multiproduct plant. The heuristic procedure used is based on deriving an initial design and then improving on it by adjusting the processing rates of the semicontinuous units, the number of batch units at each stage, and by varying the allocation of the total production time allotted to each product. However, this method does not explicitly address the location of the storage within the process. The method must be run repeatedly, with storage at different locations, in order to determine the best location for the storage.

Yeh and Reklaitis (1987) addressed the issue of storage location for a single product plant. Heuristic rules to identify the best locations for additional intermediate storage are defined; intermediate storage is located in order to maximize the equipment savings for the processing train not containing the time limiting stage. The sizing procedures for the storage are not detailed, but the bound presented by Karimi and Reklaitis (1985) can be employed.

An interesting use of intermediate storage was considered by Shah and Pantelides (1991). They note that multiproduct plants operating in campaign mode often contain products with stable intermediates. If long-term storage is available, as it is for the products, then these intermediates may be produced as the product of a given campaign. When these intermediates are used in the production of several products, the value of producing them in their own campaign is obvious. They demonstrate that even in the cases where the intermediate is only used by one product, producing it in a campaign is often beneficial because it may allow other intermediates to be produced at the same time. However, the MINLP programming formulation presented only considers such processing structures for single product plants.

Another benefit of intermediate storage is its ability to balance the effects of process fluctuations. Takamatsu *et al.* (1984) considered using intermediate storage to adjust for uncertainties in the processing times and batch sizes of the upstream and downstream units. They allowed for variations within a set of predetermined limits and determined the storage required to ensure smooth operation of the batch process.

In general, the use of intermediate storage has not been studied in a very comprehensive way in the current literature. For example, a method to locate intermediate storage in a multipurpose plant has not yet been addressed. More complicated and subjective issues such as the cost of inventory, the maintenance and clean out costs for the storage vessels, the labor costs associated with transferring material to and from the vessels, material contamination, processing delays, and the loss of batch identity have been identified but have not been dealt with in any sort of systematic fashion.

D.8 Design Under Uncertainty

The uncertainties encountered in batch process design problems can be classified into two broad categories (Johns *et al.*, 1978):

Short term Batch to batch variations. These include uncertainties in processing times, batch sizes, size factors, and equipment availability.

Long term Variations with time scales on the order of months or years. These include available production time, production amounts, and the composition of the product slate.

These two types of variations may be handled differently in the design procedure. Short term variations may be absorbed by the overdesign of individual units or storage tanks (Takamatsu *et al.*, 1984) or through the manipulation of operating conditions. Long term uncertainty is typically handled at the design stage when the engineers are faced with the decision of whether to build in the face of uncertainty or whether to wait until some uncertainty is resolved before making the decision to build or expand (Johns *et al.*, 1978).

The work of Johns *et al.* (1978) has shown that alternative design procedures should be compared based on the evaluation of objective criteria over the entire life of the plant. They noted that it is best to evaluate risk through a confidence level criterion, putting a bound on the acceptable level of risk, rather than artificially attempting to incorporate the acceptable risk level by demanding an inflated internal rate of return for the project.

Studies specifically addressing the design of a batch plants have focused on both technical and commercial uncertainty (Wellons and Reklaitis, 1989; Reinhart and Rippin, 1987; Shah and Pantelides, 1992). Wellons and Reklaitis consider uncertainties in both the design parameters and in the production quantities. To account for both types of uncertainties they partition the constraints into two sets.

hard constraints The constraints that must always be satisfied. These include mass and energy balance constraints.

soft constraints The constraints that may be violated, but the violation of these constraints would be subject to a penalty. These include ability to meet the production quantity and horizon time constraints.

The authors propose a design formulation where they insert the upper/lower bound values of the uncertain parameters into the hard constraints to insure that they are satisfied. Penalty terms are then assigned to the violation of the soft constraints. The design problem, allowing for staged expansion, is formulated as an MINLP. The authors conclusions follow those of Johns *et al.* that demonstrate it is often best to design for a planned plant expansion. In contrast to this approach, Shah and Pantelides (1992) address a set of possible production scenarios by solving a multiperiod deterministic design problem. They minimize the capital investment required to insure that the production requirements can be met for each possible set of product requirements. They employ the method of Shah and Pantelides (1991) to solve the deterministic design problem.

Short term variations and the stochastic variability have not typically been handled at the design stage. However, procedures to size intermediate storage to insure smooth operation of the plant even while batch sizes and cycle times may vary within some predetermined bounds have been addressed (Takamatsu *et al.*, 1984). However, systematic methods to account for the stochastic variation of task timings and batch size variations have not been addressed at the design stage. Felder and others have shown that the performance of a given plant subject to stochastic variations, such as uncertain processing times and equipment downtime, can be quantified by discrete event simulation (Felder, 1983; Morris, 1983; Felder *et al.*, 1985). These studies demonstrate that these variations cause the plant to perform differently from its planned mode of operation.

D.9 Solution Methods

The methods which are applicable to the batch design problem are those that can handle the optimization of both discrete and continuous variables. Most of the design problems can be formulated as a mixed integer nonlinear program (MINLP), representing the discrete decisions with integer variables. To date, no methods are available which can provide efficient solutions to such problems. In fact, in many of the cases the relaxed nonlinear program (NLP) derived from the MINLP is non-convex, so a globally optimal solution cannot be guaranteed.

Due to difficulty in solving many of the problems that might be imagined, many researchers have restricted the type of design problem that they consider. Often they choose a problem formulation that fits within the solution framework that they intend to use to solve the problem. While this technique may allow solution of the problems considered in a reasonable amount of time, these researchers may be ignoring important plant configurations which may lead to lower costs. In addition, the demands on the process may be uncertain, so obtaining the globally optimal solution to a problem that is at best an approximation is probably not all that relevant. For these reasons, it is important to consider several questions when choosing an optimization method:

- What type of problem can we formulate? How much freedom do we have in the formulation of the objective function and the constraints?
- How large a problem can we solve in a "reasonable" amount of time?
- Can we guarantee the optimality of our solution?
- How robust is our algorithm?
- How important is obtaining the optimum? Will near optimal solutions do?
- What is a "reasonable" amount of time for the problem we wish to solve?
- How easy is it to formulate the desired problem within the solution framework?

Answers to these questions should aid in the selection of an appropriate solution method for a specific problem. Some of the techniques that have been applied to the design problem will be mentioned, along with some of the advantages and disadvantages of these methods.

Sparrow *et al.* (1975) consider both heuristic procedures and branch and bound techniques to handle the discrete decisions involved in their formulation of the batch plant design. Their research demonstrated that heuristic procedures could be very effective in handling the solution of the design problem they considered. Their heuristics provided good solutions, usually optimal, in a much shorter amount of time than the branch and bound procedure. In fact, they used their heuristics to generate a feasible solution to the problem to provide a bound for the branch and bound algorithms pruning procedure.

Grossmann and Sargent (1979) show that the plant design can be formulated as an mixed integer nonlinear programming problem. Much of the research that followed considered modifications to their formulation of the problem and similar solution techniques (Suhami and Mah, 1982; Vaselenak *et al.*, 1987; Faqir and Karimi, 1989). Grossmann and Sargent, as well as others, employ variable transformations to convexify the NLP. When the relaxed NLP is convex, these methods will provide a global optimum. However, it is often difficult to derive the constraint equations to represent the process feasibility. This has been demonstrated by the difficulty in formulating the horizon time constraints for the case of the single production route multiproduct plants (Suhami and Mah, 1982; Imai and Nishida, 1984; Vaselenak *et al.*, 1987; Faqir and Karimi, 1989). These MINLP problems have been solved using both the Outer Approximation/Equality Relaxation method (Duran and Grossmann, 1986) and the Generalized Benders Decomposition (Benders, 1962; Geoffrion, 1972).

Voudouris and Grossmann (1992a) employed a different approach to attack the combinatorial nature of the design problem. They showed that some of the MINLP formulations (Grossmann and Sargent, 1979; Vaselenak *et al.*, 1987; Faqir and Karimi, 1989) can be reformulated as MILP problems. This reformulation has the advantage of being easier to solve since the relaxed problem is an LP for which efficient programming algorithms are available. In addition, the MILP algorithms are more robust. These problems can be solved more efficiently, but the objective function and constraints are fairly restrictive with respect to the type of problem that may be formulated.

An approach which circumvents restrictions on the way in which the problem is formulated was presented by Patel *et al.* (1991). They employ a simulated annealing algorithm, a type of local search technique, that enables great flexibility in the way in which they can formulate the objective function. Although they cannot guarantee optimal solutions, their method performed very well. In fact, in several example problems they obtained better solutions than were found using MINLP techniques. They were able to obtain better solutions because they could consider more complex equipment configurations than were considered in the cases solved by the MINLP methods.

D.10 Conclusions

The batch plant design problems that have been considered thus far have merely been extensions to the original work of Loonkar and Robinson (1970; 1972) and Sparrow (1975). All of the work has focused on extending the original formulations to incorporate more complex equipment superstructures and operating procedures. No attempt has been made to introduce a comprehensive reformulation of the problem, or to seriously question the assumptions inherent in the existing formulations. Hence, in his 1993 review, Rippin characterized the progress in this research as "filling in the holes."

One aspect of these formulations which needs serious consideration is the way in which restrictions and more complicated operating procedures may be considered. One step in this direction has been made by Patel *et al.* (1991) by using a simulated annealing technique. Their formulation was the first to consider parallel units of different size in a systematic procedure. The importance of allowing such operating configurations was emphasized by Flatz (1981); he considered such options when describing how a batch plant design may be improved in an evolutionary fashion. People will not be able to take advantage of these algorithms until it becomes easier to adapt these programming techniques to the real life problem facing the engineer.

The automated programming formulations for the "plant design problem" should coincide with the problem actually faced by an organization considering investment in a new manufacturing facility. Therefore, the assumptions on which these problem formulations rely need to be considered on a case by case basis. The fact that most of the solution procedures depend upon certain assumptions renders them inapplicable to many real life problems. Some aspects which deserve consideration are the cost of semicontinuous units, the cost of intermediate storage, and the cost of inventory. All of these costs have been shown to be significant in certain cases (Knopf *et al.*, 1982; Cohen and Zeftel, 1980). Although the cost associated with inventory is often difficult to assess, its cost should not be dismissed if it happens to be an important component of the total design cost. The solution methods and problem formulations must be tailored to the problem at hand, rather than attempting to tailor the problem to the desired solution method.

Finally, the fundamental premises upon which these formulations are based — fixed demand and known operating conditions — should be evaluated for their validity. We know that the life-cycle of the products is much shorter than the lifetime of the plant, so in how many cases are the demands that will be placed on the plant known with any certainty? Although a significant amount of progress has been made in addressing the design under uncertainty, the current formulations are still far from meeting the practical requirements. If the future requirements of the plant are unknown, is it worth attempting to find an optimal solution to a problem which is at best approximate and at worst irrelevant? If finding an optimal solution is worthwhile, then the most optimal operating procedures should certainly be considered at the design stage. This will add a layer of complexity and a need for advanced simulation tools, but it will create the opportunity to produce more efficient designs.

In conclusion it seems apparent that most real life batch plants are not designed in this fashion, except in the rare cases that the product demands are known within a reasonable degree of certainty. Due to the uncertainty in the demands, the ability to consider more complex operating procedures and equipment configurations is probably not the most pressing need in the area of batch design. We feel it is more appropriate to consider optimizing the design of the batch processes rather than that of the batch plants. The work on the plant design has made some progress in the area of the process design, but batch process design has been considered infrequently and is a wide open area for research.

Bibliography

W. P. Adams and H. D. Sherali. A tight linearization and an algorithm for zero-one quadratic programming problems. *Management Science*, 32(10):1274–1290, 1986.

W. P. Adams and H. D. Sherali. Linearization strategies for a class of zero-one mixed integer programming problems. *Operations Research*, 38(2):217–228, March–April 1990.

W. P. Adams and H. D. Sherali. Mixed-integer bilinear programming problems. *Mathematical Programming*, 59:279–305, 1993.

C. S. Adjiman, I. P. Androulakis, C. D. Maranas, and C. A. Floudas. A global optimization method, α bb, for process design. *Computers Chem. Engng*, 20(S):S419–S424, 1996.

B. S. Ahmad and P. I. Barton. Solvent recovery targeting for pollution prevention in pharmaceutical and specialty chemical manufacturing. *AIChE Symposium Series*, 90(303):59–73, 1994.

B. S. Ahmad and P. I. Barton. Synthesis of batch processes with integrated solvent recovery and recycling. In *Proceedings AIChE Annual Meeting*, pages Paper No. 187–D, Miami, November 1995.

B. S. Ahmad and P. I. Barton. Homogeneous multicomponent azeotropic batch distillation. *AIChE J.*, 42(12):3419–3433, December 1996.

B. S. Ahmad, Y. Zhang, and P. I. Barton. Solvent recovery targeting. *in preparation*, 1997.

B. S. Ahmad. *Synthesis of Batch Processes with Integrated Solvent Recovery.* PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, March 1997.

F. A. Al-Khayyal. Generalized bilinear programming: Part I. models, applications and linear programming relaxation. *Eur. J. Oper. Res.*, 60:306–314, 1992.

R. J. Allgor and P. I. Barton. Automatic scaling of differential-algebraic equations. *In preparation*, June 1997.

R. J. Allgor and P. I. Barton. Mixed-integer dynamic optimization. *Computers Chem. Engng*, 20(S):S451–S456, 1997.

R. J. Allgor, M. D. Barrera, P. I. Barton, and L. B. Evans. Optimal batch process development. *Computers Chem. Engng*, 20(6/7):885–896, 1996.

ANSI/IEEE Std. 754. IEEE standard for binary floating-point arithmetic. An American National Standard, Institute of Electrical and Electronics Engineers, Inc. and American National Standards Institute, New York, 1985.

M. Arioli, J. W. Demmel, and I. S. Duff. Solving sparse linear systems with sparse backward error. *SIAM J. Matrix Anal. Appl.*, 10(2):165–190, April 1989.

Aspen Technology, Inc. Achieving true potential: BATCH PLUS$^{\mathrm{TM}}$. Cambridge, MA, 1997.

AspenTech. *BATCHFRAC User Guide.* Aspen Technology, Inc., June 1991.

AspenTech. *SpeedUp User Manual Release 5.4.* Aspen Technology UK Ltd., 1993.

AspenTech. *SpeedUp Library Manual Release 5.4*, 1995.

M. J. Bagajewicz and V. Manousiouthakis. On the generalized Benders decomposition. *Computers Chem. Engng*, 15(10):691–700, October 1991.

S. Balakrishna and L. T. Biegler. Constructive targeting approaches for the synthesis of chemical reactor networks. *Ind. Eng. Chem. Res.*, 31:300–312, 1992.

S. Balakrishna and L. T. Biegler. Targeting strategies for the synthesis and energy integration of nonisothermal reactor networks. *Ind. Eng. Chem. Res.*, 31:2152–2164, 1992.

S. Balakrishna and L. T. Biegler. A unified approach for the simultaneous synthesis of reaction, energy, and separation systems. *Ind. Eng. Chem. Res.*, 32:1372–1382, 1993.

J. R. Banga and W. D. Seider. Global optimization of chemical processes using stochastic algorithms. In C. A. Floudas and P. M. Pardalos, editors, *Nonconvex Optimizaton and its Applications.* Kluwer Academic Pub., 1995.

M. D. Barrera and L. B. Evans. Optimal design and operation of batch processes. *Chem. Eng. Comm.*, pages 45–66, 1989.

M. D. Barrera. *Optimal Design and Operation of Batch Processes.* PhD thesis, Massachusetts Institute of Technology, June 1990.

P. I. Barton and S. Galán. Sensitivity features in ABACUSS. ABACUSS Project Report 97/2, Dept. of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, MA, March 1997.

P. I. Barton and C. C. Pantelides. The modelling of combined discrete/continuous processes. *AIChE Journal*, 40(6):966–979, 1994.

P. I. Barton and T. Park. Analysis and Control of Combined Discrete/Continuous Systems: Progress and Challenges in the Chemical Process Industries. *AIChE Symposium Series*, 1997. in press.

P. I. Barton, W. F. Feehery, and J. E. Tolsma. Optimization features in ABACUSS. ABACUSS Project Report 96/1, Dept. of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, MA, January 1996.

P. I. Barton. *The Modelling and Simulation of Combined Discrete/Continuous Processes.* PhD thesis, University of London, 1992.

P. I. Barton. Batch process simulation: Why and how. In *AspenWorld 1994*, Boston, Massachusetts, November 6–9 1994.

P. I. Barton. Generalized junction conditions for parametric sensitivities. ABACUSS Project Report 96/3, Dept. of Chemical Engineering, Massachusetts Institute of Technology, Cambridge, MA, December 1996.

F. L. Bauer, J. Stoer, and C. Witzgall. Absolute and monotonic norms. *Numer. Math.*, 3:257–264, 1961.

F. L. Bauer. Optimally scaled matrices. *Numer. Math.*, 5:73–87, 1963.

E. M. L. Beale and J. A. Tomlin. Special facilities in a general mathematical programming system for non-convex problems using ordered sets of variables. In J. Lawrence, editor, *OR 69: Proceedings of the Fifth International Converence on Operational Research*, pages 447–454, New York, 1970. International Federation of Operational Research Societies, Tavistock.

J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252, 1962.

C. Bernot, M. F. Doherty, and M. F. Malone. Patterns of composition change in multicomponent batch distillation. *Chemical Eng. Science*, 45(5):1207, 1990.

T. Bhatia and L. T. Biegler. Dynamic optimization in the design and scheduling of multiproduct batch plants. *Ind. Eng. Chem. Res.*, 35(7):2234, 1996.

D. B. Birewar and I. E. Grossmann. Incorporating scheduling in the optimal design of multiproduct batch plants. *Computers Chem. Engng.*, 13(1/2):141–161, 1989.

C. Bischof, A. Carle, G. Corliss, A. Griewank, and P. Hovland. ADIFOR: Generating derivative codes from Fortran programs. *Scientific Programming*, 1(1):1–29, 1992.

J. Blazewics, M. Dror, and J. Weglarz. Mahematical programming formulations for machine scheduling: a survey. *Eur. J. Oper. Res.*, 51:283–300, 1991.

J. P. Boston, H. J. Britt, S. Jirapongphan, and V. B. Shah. BATCHFRAC: An advanced system for simulation of batch distillation operations. In R. S. H. Mah and W. D. Seider, editors, *Proceedings of the Conference on Foundations of Computer-Aided Process Design*, volume 2, page 203, New York, July 1981. Engineering Foundation.

R. D. Braatz and M. Morari. Minimizing the Euclidean conditon number. *SIAM J. Control and Optimization*, 32(6):1763–1768, November 1994.

R. W. Brankin, I. Gladwell, and L. F. Shampine. Starting BDF and Adams codes at optimal order. *J. Comp. Appl. Math.*, 21(3):357–368, March 1988.

K. E. Brenan, S. L. Campbell, and L. R. Petzold. *Numerical Solution of Initial-Value Problems in Differential Algebraic Equations*. SIAM, Philadelphia, 1996.

A. Brooke, D. Kendrick, and A. Meeraus. *GAMS A User's Guide Release 2.25.* The Scientific Press, San Francisco, CA, 1992.

L. Brüll and U. Pallaske. On consistent initialization of differential-algebraic equations with discontinuities. In H. Wacker and W. Zulehner, editors, *Proceedings of the Fourth Equopean conference on Mathematics in Industry*, pages 213–217, Netherlands, 1991. B. G. Teubner Stuttgart and Kluwer Academic Publishers.

L. Brüll and U. Pallaske. On differential algebraic equations with discontinuities. *Z angew Math Phys*, 43:319–327, 1992.

C. Bruneton, C. Hoff, and P. I. Barton. Computer aided identification of chemical reaction hazards. *Computers Chem. Engng*, 20(S):S 311–S318, 1997.

A. E. Bryson and Y. Ho. *Applied Optimal Control*. Hemisphere, New York, 1975.

Pawel Bujakiewicz. *Maximum weighted matching for high index differential algebraic equations*. PhD thesis, TU Delft, Delft, Netherlands, 1994.

F. E. Cellier and H. Elmqvist. Automated formula manipulation supports object-oriented continuous-system modeling. *IEEE Control Systems Magazine*, 13(2):28–38, April 1993.

F. E. Cellier. *Combined Continuous/Discrete System Simulation by Use of Digital Computers: Techniques and Tools*. PhD thesis, Swiss Federal Institute of Technology Zurich, 1979.

M. S. Charalambides, N. Shah, and C. C. Pantelides. Optimal batch process synthesis. In *Proceedings AIChE Annual Meeting*, pages Paper No. 153–C, St. Louis, November 1993.

M. S. Charalambides, N. Shah, and C. C. Pantelides. Synthesis of batch reaction/distillation processes using detailed dynamic models. *Computers Chem. Engng*, 19(S):S167–S174, 1995.

M. S. Charalambides, N. Shah, and C. C. Pantelides. Optimal design of integrated batch processes. I. preliminary process design. In *Proceedings AIChE Annual Meeting*, pages Paper No. 190–G, Miami, November 1995.

M. S. Charalambides. *Optimal Design of Integrated Batch Processes*. PhD thesis, University of London, November 1996.

Y. Chung and A. W. Westerberg. A proposed numerical algorithm for solving nonlinear index problems. *Ind. Eng. Chem. Res.*, 29(7):1234–1239, 1990.

Y. Chung and A. W. Westerberg. Solving a class of near index problems as perturbations of index problems. *Ind. Eng. Chem. Res.*, 31(11):2593–2603, 1992.

R. L. Cohen and L. Zeftel. Materials requirements planning in the CPI. *CEP*, pages 59–63, April 1980.

CPLEX Optimization, Inc., Incline Village, NV. *Using the CPLEX Callable Library and CPLEX Mixed Integer Library*, 1993.

E. W. Crabtree and M. M. El-Halwagi. Synthesis of acceptable reactions. *AIChE Symposium Series*, 90(303):117–127, 1994.

C. A. Crooks and S. Macchietto. A combined MILP and logic-based approach to the synthesis of operating procedures for batch plants. *Chem. Eng. Comm.*, 114:117–144, 1992.

J. E. Cuthrell and L. T. Biegler. On the optimization of differential-algebraic process systems. *AIChE J.*, 33:1257–1270, 1987.

J. E. Cuthrell and L. T. Biegler. Simultaneous optimization and solution methods for batch reactor control profiles. *Computers Chem. Engng*, 13(1–2):49–62, 1989.

A. J. Czulek. An experimental simulator for batch chemical processes. *Computers Chem. Engng*, 12(2/3):253–259, 1988.

M. M. Daichendt and I. E. Grossmann. Preliminary screening for the MINLP synthesis of process systems — I. Aggregation and decomposition techniques. *Computers Chem. Engng.*, 18(8):663–678, 1994.

M. M. Daichendt and I. E. Grossmann. Preliminary screening procedure for the MINLP synthesis of process systems — II. Heat exchanger networks. *Computers Chem. Engng.*, 18(8):679–709, 1994.

A. G. Davidyan, V. N. Kiva, G. A. Meski, and M. Morari. Batch distillation in a column with a middle vessel. *Chemical Engineering Science*, 49(18):3033–3051, 1994.

K. G. Denbigh. Optimum temperature sequences in reactors. *Chem. Eng. Sci.*, 8:125–132, 1958.

Urmila M. Diwekar. *Batch Distillation : Simulation, Optimal design, and Control.* Series in chemical and mechanical engineering. Taylor and Francis, Washington, DC, 1995.

M. F. Doherty and J. D. Perkins. On the dynamics of distillation processes - I: The simple distillation of multicomponent non-reacting, homogeneous liquid mixtures. *Chem. Eng. Science*, 33:281–301, 1978.

M. F. Doherty and J. D. Perkins. On the dynamics of distillation processes - II: The simple distillation of model solutions. *Chem. Eng. Science*, 33:569–578, 1978.

M. F. Doherty and J. D. Perkins. On the dynamics of distillation processes - III: The topology structure of ternary residue curve maps. *Chem. Eng. Science*, 34:1401–1414, 1979.

I. S. Duff and J. K. Reid. MA48, a fortran code for direct solution of sparse unsymmetric linear systems of equations. Technical Report RAL-93-072, Rutherford Appleton Laboratory, Oxon, UK, October 1993.

I. S. Duff and J. K. Reid. The design of MA48, a code for the direct solution of sparse unsymmetric linear systems of equations. Technical Report RAL-TR-95-039, Rutherford Appleton Laboratory, Oxon, UK, August 1995.

I. S. Duff, A. M. Erisman, and J. K. Reid. *Direct Methods for Sparse Matrices.* Clarendon Press, Oxford, 1986.

M. A. Duran and I. E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming*, 36:307–339, 1986.

A. Edelman. The complete pivoting conjecture for Gaussian elimination is false. *Mathematica Journal*, 2:58–61, 1992.

D. Ellison. Efficient automatic integration of ordinary differential equations with discontinuities. *Mathematics and Computers in Simulation*, 23:12–20, 1981.

L. B. Evans. Steady-state simulation: A state-of-the-art review. In *The Fifth International Symposium on Process Systems Engineering*, pages 953–967, Kyongju, Korea, July 1994.

J. R. Fair, D. E. Steinmeyer, W. R. Penney, and B. B. Crocker. Liquid-gas systems. In R. H. Perry, D. W. Green, and J. O. Maloney, editors, *Perry's Chemical Engineers' Handbook Sixth Edition*, pages 18.1–18.88. McGraw-Hill, Inc., New York, 1984.

N. M. Faqir and I. A. Karimi. Optimal design of batch plants with single production routes. *Ind. Eng. Chem. Res.*, 28(8):1191–1202, 1989.

N. M. Faqir and I. A. Karimi. Design of multipurpose batch plants with multiple production routes. In *Proceedings Third International Conference on Foundations of Computer-Aided Process Design*, pages 451–468, Snowmass, Colorado, 1990.

W. F. Feehery and P. I. Barton. A novel approach to dynamic optimization of ODE and DAE systems as high index problems. In *AIChE Annual Meeting*, Miami Beach, Florida, November 1995.

W. F. Feehery and P. I. Barton. A differentiation-based approach to simulation and dynamic optimization with high-index differential-algebraic equations. In M. Berz, C. Bischof, G. Corliss, and A. Griewank, editors, *Computational Differentiation.* SIAM, 1996.

W. F. Feehery and P. I. Barton. Dynamic simulation and optimization with inequality path constraints. *Computers Chem. Engng*, 20(S):S707–S712, 1996.

W. F. Feehery, J. E. Tolsma, and P. I. Barton. Efficient sensitivity analysis of large-scale differential-algebraic systems. *Appl. Num. Math.*, May 1997. in press.

R. M. Felder, G. B. McLeod, and R. F. Moldin. Simulation for the capacity planning of specialty chemicals production. *CEP*, pages 41–46, June 1985.

R. M. Felder. Simulation — a tool for optimizing batch-process production. *Chemical Engineering*, pages 79–84, April 18, 1983.

Z. T. Fidkowski, M. F. Malone, and M. F. Doherty. Computing azeotropes in multicomponent mixtures. *Computers Chem. Engng*, 17(12):1141–1155, 1993.

W. Flatz. Equipment sizing for multiproduct plants. *Chemical Engineering*, pages 71–80, February 25, 1980.

W. Flatz. Sizing equipment for multiproduct plants. *Chemical Engineering*, pages 105–115, July 13, 1981.

Christodoulos A. Floudas. *Nonlinear and Mixed-Integer Optimization.* Oxford University Press, New York, 1995.

G. E. Forsythe and C. B. Moler. *Computer Solution of Linear Algebraic Systems.* Prentice-Hall, Inc., Englewood Cliffs, NJ, 1967.

N. R. Foulkes, M. J. Walton, P. K. Andow, and M. Galluzzo. Computer-aided synthesis of complex pump and valve operations. *Computers Chem. Engng*, 12(9/10):1035–1044, 1988.

W. M. Fruit, G. V. Reklaitis, and J. M. Woods. Simulation of multiproduct batch chemical processes. *The Chemical Engineering Journal*, 8:199–211, 1974.

C. W. Gear. Simultaneous numerical solution of differential-algebraic equations. *IEEE Transactions on Circuit Theory*, CT-18:89–95, 1971.

C. W. Gear. Method and initial stepsize selection in multistep ODE solvers. Technical Report UIUCDCS-R-80-1006, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, 1980.

C. W. Gear. Runge-Kutta starters for multistep methods. *ACM Trans. Math. Software*, 6:263–279, 1980.

A. M. Geoffrion. Generalized Benders decomposition. *Journal of Optimization Theory and Applications*, 10:237–260, 1972.

I. Gladwell, L. F. Shampine, and R. W. Brankin. Automatic selection of the initial step size for an ODE solver. *J. Comp. Appl. Math.*, 18(2):175–192, 1987.

I. Gladwell. Initial value routines in the NAG library. *ACM Trans. Math. Software*, 5:386–400, 1979.

B. Glasser, D. Hildebrandt, and D. Glasser. Optimal mixing for exothermic reversible reactions. *Ind. Eng. Chem. Res.*, 31:1541–1549, 1992.

F. Glover and E. Wolsey. Converting the 0–1 polynomial programming problem to a 0–1 linear program. *Operations Research*, 22:180–182, 1974.

F. Glover. Improved linear integer programming formulations of nonlinear integer problems. *Mgmt. Sci.*, 22:455–460, 1975.

G. H. Golub and C. F. Van Loan. *Matrix Computations.* Johns Hopkins University Press, Baltimore, second edition, 1989.

N. Gould. On growth in Gaussian elimination with complete pivoting. *SIAM J. Matrix Anal. Appl.*, 12:354–361, 1991.

I. E. Grossmann and R. W. H. Sargent. Optimum design of multipurpose chemical plants. *Ind. Eng. Chem. Process Des. Dev.*, 18(2):343–348, 1979.

I. E. Grossmann, V. T. Voudouris, and O. Ghattas. Mixed-integer linear programming reformulations for some nonlinear discrete design optimization problems. In C. A. Floudas and P. M. Pardalos, editors, *Recent Advances in Global Optimization*, pages 478–512. Princeton University Press, 1992.

H. Guggenheimer, A. S. Edelman, and C. R. Johnson. A simple estimate fo the condition number of a linear system. *College Mathematics Journal*, 26(1):3–5, 1995.

E. Hairer and G. Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Equations*, volume 14 of *Springer Series in Computational Mathematics.* Springer-Verlag, New York, 1991.

E. Hairer and G. Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*, volume 8 of *Springer Series in Computational Mathematics.* Springer-Verlag, New York, second edition, 1993.

Harwell. *Harwell Subroutine Library - Specifications (Release 11): Volume 1.* Theoretical Studies Department, AEA Technology, Didcot, Oxfordhsire, UK, June 1993.

T. Hatipoglu and D. W. T. Rippin. The effects of optimal temperature and feed addition rate profiles on batch reactions and sequences of batch operations — the development of a catalogue of attainable regions. In *Institution of Chemical Engineers Symposium Series No. 87*, pages 447–454. Pergamon Press, 1984.

A. C. Hearn. *REDUCE User's Manual, Version 3.3.* The Rand Corporation, Santa Monica, CA, 1987.

N. J. Higham and D. J. Higham. Large growth factors in Gaussian elimination with pivoting. *SIAM J. Matrix Anal. Appl.*, 10(2):155–164, April 1989.

Nicholas J. Higham. *Accuracy and Stability of Numerical Algorithms.* SIAM, Philadelphia, 1996.

D. Hildebrandt and L. T. Biegler. Synthesis of chemical reactor networks. In L. T. Biegler and M. F. Doherty, editors, *Fourth International Conference on Foundations of Computer-Aided Process Design*, pages 52–67, Snowmass, CO, July 1995. CACHE.

D. Hildebrandt and D. Glasser. The attainable region and optimal reactor structures. *Chem. Eng. Science*, 45(8):2161–2168, 1990.

D. Hildebrandt, D. Glasser, and C. Crowe. Geometry of the attainable region generated by reaction and mixing: With and without constraints. *Ind. Eng. Chem. Res.*, 29:49–58, 1990.

K. E. Hillstrom. Users guide for JAKEF. Technical Report Technical Memorandum ANL/MCS-TM-16, Mathematics and Computer Science Division, Argonne National Laboratory, 9700 South Cass Ave., Argonne, IL 60439, 1985.

IBM Corporation. *Optimization Subroutine Library Guide and Reference*, SC23–0519–2 1991.

M. Imai and N. Nishida. New procudure generating suboptimal configurations to the optimal design of multipurpose batch plants. *Ind. Eng. Chem. Process Des. Dev.*, 23(4):845–847, 1984.

M. Iri and K. Kubota. Norms, rounding errors, partial derivatives and fast automatic differentiation. *IEICE Transactions*, E74(3):463–471, March 1991.

M. Iri, T. Tsuchiya, and M. Hoshi. Automatic computation of partial derivatives and rounding error estimates with applications to large-scale systems of nonlinear equations. *J. Comp. Appl. Math.*, 24(3):365–392, December 1988.

M. Iri. History of automatic differentiation and rounding error estimation. In A. Griewank and G. F. Corliss, editors, *Automatic Differentiation of Algorithms: Theory, Implementation, and Application*, chapter One, pages 3–16. SIAM, 1988.

R. B. Jarvis and C. C. Pantelides. Numerical methods for differential-algebraic equations new problems and new solutions. In *Proceedings AIChE Annual Meeting*, page Session 158, Los Angeles, November 18–22 1991.

R. B. Jarvis and C. C. Pantelides. DASOLV — a differential algebraic equation solver. Technical report, Centre for Process Systems Engineering, Imperial College, London, March 1992.

S. Jayakumar and G. V. Reklaitis. Chemical plant layout via graph partitioning–I. Single level. *Computers Chem. Engng.*, 18(5):441, May 1994.

S. Jayakumar and G. V. Reklaitis. Chemical plant layout via graph partitioning-II. Multiple levels. *Computers Chem. Engng.*, 20(5):563–578, 1996.

G. S. Joglekar and G. V. Reklaitis. A simulator for batch and semi-continuous processes. *Computers Chem. Engng*, 8(6):315–327, 1984.

W. R. Johns, G. Marketos, and D. W. T. Rippin. The optimal desing of chemical plant to meet time-varying demands in the presence of technical and commercial uncertainty. *Trans IChemE*, 56:249–257, 1978.

I. A. Karimi and G. V. Reklaitis. Optimal selection of intermediate storage tank capacity in a periodic batch/semicontinuous process. *AIChE Journal*, 29(4):588–596, 1983.

I. A. Karimi and G. V. Reklaitis. Intermediate storage in noncontinuous processes involving stages of parallel units. *AIChE Journal*, 31(1):44–52, 1985.

B. J. Keeping. *Efficient methods for the Solution of Large Systems of Differential-Algebraic Equations.* PhD thesis, University of London, July 1995.

S. E. Ketner. Minimize batch equipment cost. *Chemical Engineering*, pages 121–124, 22 August 1960.

L. M. Kiraly, F. Friedler, and L. Szoboszlai. Optimal design of multi-purpose batch chemical plants. *Computers Chem. Engng.*, 13(4/5):527–534, 1989.

Henry Z. Kister. *Distillation Operation.* McGraw-Hill, Inc., New York, 1990.

Henry Z. Kister. *Distillation Design.* McGraw-Hill, Inc., New York, 1992.

J. P. Knight and G. J. McRae. An approach to process integration based on the choice of system chemistry. *Presented at AIChE Annual Meeting: Paper No. 153d*, November 1993.

J. P. Knight, M. A. Tatang, and G. J. McRae. Incorporating environmental objectives in process design and operation. In *Proceedings of second International Conferance on Foundations of Computer-Aided Process Operations*, Crested Butte, Colorado, July 18–23 1993.

F. C. Knopf, M. R. Okos, and G. V. Reklaitis. Optimal design of batch/semicontinous processes. *Ind. Eng. Chem. Process Des. Dev.*, 21(1):79–86, 1982.

E. Kondili, C. C. Pantelides, and R. W. H. Sargent. A general algorithm for scheduling of batch operations. In *Proceedings Third International Symposium on Process Systems Engineering*, pages 62–75, Sydney, Australia, 1988.

E. Kondili, C. C. Pantelides, and R. W. H. Sargent. A general algorithm for short-term scheduling of batch operations — I. MILP formulation. *Computers Chem. Engng*, 17(2):211, 1993.

D. Kraft. On converting optimal control problems into nonlinear programming problems. *Comp. Math. Prog.*, 15:261–280, 1985.

A. Kröner, W. Marquardt, and E. D. Gilles. Computing consistent initial conditions for differential-algebraic equations. *Computers Chem. Engng.*, 16:S131–S138, 1992.

K. Kubota and M. Iri. Estimates of rounding errors with fast automatic differentiation and interval analysis. *Journal of Information Processing*, 14(4):508–515, 1991.

R. Lakshmanan and G. Stephanopoulos. Synthesis of operating procedures for complete chemical plants — III. planning in the presence of qualitative, mixing constraints. *Computers Chem. Engng*, 14:301–317, 1990.

J. D. Lambert. *Numerical Methods for Ordinary Differential Systems: the initial value problem*. John Wiley and Sons, Inc., New York, 1991.

A. A. Linninger, S. A. Ali, and G. Stephanopoulos. Knowledge-based validation and waste management of batch pharmaceutical process designs. *Computers Chem. Engng*, 20:S1431–S1436, S 1996.

A. A. Linninger, E. Stephanopoulos, S. A. Ali, A. Salomone, A. Modi, J. Aumond, and G. Stephanopoulos. BatchDesign-Kit: A comprehensive computer-aided design framework with ecological considerations. In *Proceedings AIChE Annual Meeting*, Chicago, IL, November 1996. AIChE.

J. S. Logsdon and L. T. Biegler. Accurate solution of differential-algebraic optimization problems. *Ind. Eng. Chem. Res.*, 28:1628–1639, 1989.

Y. R. Loonkar and J. D. Robinson. Minimization of capital investment for batch processes. *Ind. Eng. Chem. Process Des. Dev.*, 9(4):625–629, 1970.

S. Macchietto. Bridging the gap — integration of design, operations scheduling and control. In *Proceedings of Second International Conference on Foundations of Computer-Aided Process Operations*, Crested Butte, Colorado, July 18–23, 1993.

T. Maly and L. R. Petzold. Numerical methods and software for sensitivity analysis of differential-algebraic systems. *Appl. Num. Math.*, 20:57–79, 1996.

J. A. Mandler. Implementation issues in the dynamic simulation of complex chemical processes. In *Proceedings AIChE Annual Meeting*, Miami, FL, November 1992.

C. D. Maranas and C. A. Floudas. Global optimization in generalized geometric programming. *Computers Chem. Engng*, 21(4):351–369, 1996.

S. E. Mattsson and G. Söderlind. Index reduction in differential-algebraic equations using dummy derivatives. *SIAM J. Sci. and Stat. Comp.*, 14(3):677–692, 1993.

S. E. Mattsson. On modeling and differential/algebraic systems. *Simulation*, 52(1):24–32, January 1989.

G. P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I: — convex underestimating problems. *Mathematical Programming*, 10:146–175, 1976.

A. K. Modi and I. A. Karimi. Design of multiproduct batch processes with finite intermediate storage. *Computers Chem. Engng.*, 13(1/2):127–139, 1989.

A. Modi, J. P. Aumond, M. Mavrovouniotis, and G. Stephanopoulos. Rapid plant-wide screening of solvents for batch processes. *Computers Chem. Engng*, 20(S):S375–S380, 1996.

M. J. Mohideen, J. D. Perkins, and E. N. Pistikopoulos. Optimal synthesis and design of dynamic systems under uncertainty. *Computers Chem. Engng.*, 20(S):S895–S900, 1996.

J. M. Montagna, O. A. Iribarren, and F. C. Galiano. The design of multiproduct batch plants with process performance models. *Trans. Inst. Chem. Eng.*, 72,(6):783–791, 1994.

J. J. Moré and D. C. Sorensen. Newton's method. In G. H. Golub, editor, *Studies in Numerical Analysis*, pages 29–83. Mathematical Association of America, Washington, D.C., 1984.

K. R. Morison and R. W. H. Sargent. Optimization of multistage processes described by differential-algebraic equations. *Lect. Notes Math.*, 1230:86–102, 1986.

R. C. Morris. Simulating batch processes. *Chemical Engineering*, pages 77–81, May 16, 1983.

I. M. Mujtaba and S. Macchietto. The role of holdup on the performance of binary batch distillation. In *Proceedings Fourth International Symposium on Process Systems Engineering*, pages I.19.1–I.19.15, Montebello, Canada, August 5–9 1991.

I. M. Mujtaba and S. Macchietto. Optimal operation of multicomponent batch distillation – multiperiod formulation and solution. *Computers Chem. Engng*, 17(12):1191–1207, 1993.

L. Naess, A. Mjaavatten, and J. O. Li. Using dynamic process simulation from conception to normal operation of process plants. *Computers Chem. Engng.*, 17:585–600, 1993.

S. Ochs, P. Rosendorf, I. Hyanek, Z. Xiaomang, and W. H. Ray. Dynamic flowsheet modeling of polymerization processes using POLYRED. *Computers Chem. Engng.*, 20(6/7):657–663, 1996.

C. C. Pantelides and P. I. Barton. Equation-oriented dynamic simulation current status and future perspectives. *Computers Chem. Engng*, 17:S263–S285, 1993.

C. C. Pantelides, D. Gritsis, K. R. Morison, and R. W. H. Sargent. The mathematical modeling of transient systems using differential-algebraic equations. *Computers Chem. Engng.*, 12(5):449–454, 1988.

C. C. Pantelides. The consistent initialization of differential-algebraic systems. *SIAM J. Sci. and Stat. Comp.*, 9:213–231, 1988.

C. C. Pantelides. Unified frameworks for optimal process planning and scheduling. In *Proceedings of Second International Conference on Foundations of Computer-Aided Process Operations*, Crested Butte, CO, July 1993.

Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization : algorithms and complexity.* Prentice Hall, Englewood Cliffs, N.J., 1982.

S. Papageorgaki and G. V. Reklaitis. Optimal design of multipurpose batch plants. 1. Problem formulation. *Ind. Eng. Chem. Res.*, 29(10):2054–2062, 1990.

S. Papageorgaki and G. V. Reklaitis. Optimal design of multipurpose batch plants. 2. A decompostion solution strategy. *Ind. Eng. Chem. Res.*, 29(10):2062–2073, 1990.

T. Park and P. I. Barton. State event location in differential-algebraic models. *ACM Trans. Mod. Comput. Sim.*, 6(2):137–165, 1996.

A. N. Patel, R. S. H. Mah, and I. A. Karimi. Preliminary design of multiproduct noncontinuous plants using simulated annealing. *Computers Chem. Engng.*, 15(7):451–469, 1991.

J. F. Pekny and M. G. Zentner. Learning to solve process scheduling problems: The role of rigorous knowledge acquisition frameworks. In *Proceedings of Second International Conference on Foundations of Computer-Aided Process Operations*, Crested Butte, CO, July 1993.

L. R. Petzold. A description of DASSL: a differential/algebraic system solver. In *IMACS World Congress on System Simulation and Scientific Computation*, pages 430–432, Montreal, Canada, August 1982. IMACS.

L. R. Petzold. Differential/algebraic equations are not ODEs. *SIAM J. Sci. and Stat. Comp.*, 3:367–384, 1982.

J. M. Pinto and I. E. Grossmann. A continuous time mixed integer linear programming model for short term scheduling of multistage batch plants. *Ind. Eng. Chem. Res.*, 34:3037–3051, 1995.

E. Polastro and J. Nystrom. The fine chemicals industry – the exodus outside the triad? In *Beating the Recession: Chemical Specialities USA '93 Symposium*, pages 5–8, Pittsburgh, PA, September 27–28 1993. Spring Innovations Limited.

L. S. Pontryagin, V. Boltyanskii, R. Gamkrelidge, and E. Mishenko. *The Mathematical Theory of Optimal Processes.* Interscience Publishers Inc., New York, 1962.

A. A. B. Pritsker and R. E. Hurst. GASP IV: a combined continuous-discrete FORTRAN-based simulation language. *Simulation*, 21:65–70, 1973.

A. A. B. Pritsker. *Introduction to Simulation and SLAM II.* John Wiley, 1986.

I. Quesada and I. E. Grossmann. Global otimization of bilinear processs networks with multicomponent flows. *Computers Chem. Engng.*, 19(12):1219–1242, 1995.

H. W. Ray. Dynamic modelling of polymerization process flowsheets using POLYRED. In *1st Industrial Chemical Engineering Technology Topical Conference, AIChE Annual Meeting*, St. Louis, USA, November 7–12 1993.

R. C. Reid, J. M. Prausnitz, and B. E. Poling. *The Properties of Gases and Liquids, 4th ed.* McGraw-Hill, Inc, New York, 1987.

J. K. Reid. Sparse matrices. In A. Iserles and M. J. D. Powell, editors, *The State of the Art in Numerical Analysis*, pages 59–85. Oxford University Press, Oxford, UK, 1987.

H. J. Reinhart and D. W. T. Rippin. Design of flexible multi-product plants — a new procedure for optimal equipment sizing under uncertainty. In *Proceedings AIChE Annual Meeting*, New York, November 1987.

G. V. Reklaitis and M. L. Preston. A perspective on computer integrated process engineering. In *IChemE Symposium Series No. 114 — Computer Integrated Process Engineering: CIPE '89*, pages 315–331. Hemisphere Publishing, Corp, New York, 1989.

G. V. Reklaitis. Progress and issues in computer aided batch process design. In *Proceedings Third International Conference on Foundations of Computer-Aided Process Design*, pages 241–276, Snowmass, Colorado, 1989.

G. V. Reklaitis. Perspectives on scheduling and planning of process operations. In *Proceedings Fourth International Symposium on Process Systems Engineering*, Montebello, Canada, 1991.

G. V. Reklaitis. Overview of scheduling and planning of batch process operations. In *Proceedings NATO Advanced Study Institute — Batch Process Systems Engineering*, Antayla, Turkey, June 1992. NATO Advanced Study Institute.

D. W. T. Rippin. Design and operation of multiproduct and multipurpose batch chemical plants — an analysis of problem structure. *Computers Chem. Engng.*, 7(4):463–481, 1983.

D. W. T. Rippin. Simulation of single and multiproduct batch chemical plants for optimal design and operation. *Computers Chem. Engng.*, 7(3):137–156, 1983.

D. W. T. Rippin. Batch process systems engineering: A retrospective and and prospective review. *Computers Chem. Engng.*, 17(S):S1–S13, 1993.

J. R. Rivas and D. F. Rudd. Synthesis of failure-safe operations. *AIChE J.*, 20(2):320–325, March 1974.

J. D. Robinson and Y. R. Loonkar. Minimising capital investment for multi-product batch-plants. *Process Technology International*, 17(11):861–863, November 1972.

N. V. Sahinidis and I. E. Grossmann. Convergence properties of generalized Benders decomposition. *Computers Chem. Engng*, 15(7):481–491, July 1991.

H. E. Salomone and O. A. Iribarren. Posynomial modeling of batch plants: A procedure to include process decision variables. *Computers Chem. Engng*, 16(3):173–184, 1992.

H. E. Salomone, J. M. Montagna, and O. A. Iribarren. Dynamic simulations in the design of batch processes. *Computers Chem. Engng*, 18(3):191–204, 1994.

R. W. H. Sargent and G. R. Sullivan. The development of an efficient optimal control package. In *Proc. 8 th IFIP Conf. Optimization Tech. Pt. 2*, 1977.

G. Schilling and C. C. Pantelides. A simple continuous-time process scheduling formulation and a novel solution algorithm. *Computers Chem. Engng*, 20(S):S1221–1226, 1996.

D. Sedes. Modeling, simulation and process safety analysis. A case study: the formaldehyde process. Technical report, Department of Chemical Engineering, MIT, 1995.

A. E. Sedgwick. An effective variable order Adams method. Technical Report 53, Dept. of Computer Science, University of Toronto, Toronto, Canada, 1973.

N. Shah and C. C. Pantelides. Optimal long-term campaign planning and design of batch operations. *Ind. Eng. Chem. Res.*, 30(10):2308–2321, 1991.

N. Shah and C. C. Pantelides. Design of multipurpose batch plants with uncertain production requirements. *Ind. Eng. Chem. Res.*, 31(5):1325–1337, 1992.

N. Shah, C. C. Pantelides, and R. W. H. Sargent. A general algorithm for short-term scheduling of batch operations — II. Computational issues. *Computers Chem. Engng*, 17(2):229, 1993.

N. Shah. *Efficient Scheduling, Planning and Design of Multipurpose Batch Plants.* PhD thesis, Imperial College, February 1992.

L. F. Shampine and M. K. Gordon. Some numerical experiments with DIFSUB. *SIGNUM Newsletter*, 7:24–26, 1972.

L. F. Shampine. Lipschitz constants and robust ODE codes. In J. T. Oden, editor, *Computational Methods in Nonlinear Mechanics*, chapter 19, pages 427–449. North-Holland, New York, 1980. Proceedings of the TICOM Second International Conference.

L. F. Shampine. Measuring stiffness. *Applied Numerical Mathematics*, 1(2):107–119, March 1985.

L. F. Shampine. Starting variable-order Adams and BDF codes. *Applied Numerical Mathematics*, 3(4):331–337, August 1987.

L. F. Shampine. Ill-conditioned matrices and the integration of stiff ODEs. *J. Comp. Appl. Math.*, 48:279–292, 1993.

Shell. *Shell Briefing Service Report on Petrochemicals.* Shell Briefing Service, 1990.

R. J. W. Sim. CADSIM – user's guide and reference manual. Technical report, Imperial College, 1975.

E. M. B. Smith and C. C. Pantelides. Global optimisation of general process models. In I. E. Grossmann, editor, *Global Optimization in Engineering Design*, chapter 12. Kluwer, Boston, MA, December 1995. in press.

E. L. Sørensen, S. W. Sørensen, and R. Gani. Simulation and analysis of batch chemical processes. In L. Puigjaner and A. Espuña, editors, *Computer–Oriented Process Engineering.* Elsevier, 1991.

R. E. Sparrow, G. J. Forder, and D. W. T. Rippin. The choice of equipment sizes for multiproduct batch plants. Heuristics vs. branch and bound. *Ind. Eng. Chem. Process Des. Dev.*, 14(3):197–203, 1975.

G. W. Stewart and J. Sun. *Matrix Perturbation Theory.* Academic Press, Inc., New York, 1990.

S. C. Stinson. Custom chemicals. *C&EN*, pages 34–59, February 8, 1993.

G. Strang. *Linear Algebra and Its Applications.* Academic Press, New York, second edition, 1980.

I. Suhami and R. S. H. Mah. Optimal design of multipurpose batch plants. *Ind. Eng. Chem. Process Des. Dev.*, 21(1):94–100, 1982.

E. B. Sund and K. Lien. An optimal control formulation of the reaction-mixing problem. In *Proceedings AIChE Annual Meeting*, Chicago, November 1996.

S. Sundaram and L. B. Evans. Synthesis of separations by batch distillation. *Ind. Eng. Chem. Res.*, 32(3):500–510, March 1993.

T. Takamatsu, I. Hashimoto, and S. Hasebe. Optimal design and operation of a batch process with intermediate storage tanks. *Ind. Eng. Chem. Process Des. Dev.*, 21(3):431–440, 1982.

T. Takamatsu, I. Hashimoto, S. Hasebe, and M. O'Shima. Design of a flexible batch process with intermediate storage tanks. *Ind. Eng. Chem. Process Des. Dev.*, 23(1):40–48, 1984.

S. Tan, R. S. H. Mah, and I. A. Karimi. An interactive approach to the design of noncontinuous plants. *Computers Chem. Engng.*, 17(1):71–93, 1993.

P. Tanartkit and L. T. Biegler. Stable decomposition for dynamic optimization. *Ind. Eng. Chem. Res.*, 34:1253–1266, 1995.

Menner A. Tatang. *Direct incorporation of uncertainty in chemical and environmental engineering systems.* PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, 1995.

D. M. Terry, A. J. Zehnder, C. F. Cordeiro, and G. S. Joglekar. Using BATCHES to evaluate process alternatives through resource management. In *Proceedings AIChE Spring National Meeting*, Houston, TX, April 1989.

J. E. Tolsma and P. I. Barton. On computational derivatives. *Submitted to Comput. Chem. Engng.*, 1997.

B. Tricoire. *Design and scheduling of multiproduct batch plants with application to polymer production.* PhD thesis, University of Massachusetts, Amherst, MA, 1992.

A. van der Sluis. Condition numbers and equilibration of matrices. *Numer. Math.*, 14:14–23, 1969.

A. van der Sluis. Condition, eqilibration and pivoting in linear algebraic systems. *Numer. Math.*, 15:74–86, 1970.

D. B. Van Dongen and M. F. Doherty. On the dynamic of distillation processes-VI: Batch distillation. *Chem. Eng. Science*, 40:2087–2093, 1985.

S. Vasantharajan and L.T. Biegler. Simultaneous strategies for optimization of differential-algebraic systems with enforcement of error criteria. *Computers Chem. Engng*, 14:1083–1100, 1990.

J. A. Vaselenak, I. E. Grossmann, and A. W. Westerberg. An embedding formulation for the optimal scheduling and design of multipurpose batch plants. *Ind. Eng. Chem. Res.*, 26(1):139–148, 1987.

V. S. Vassiliadis, C. C. Pantelides, and R. W. H. Sargent. Solution of a class of multistage dynamic optimization problems. 1. problems without path constraints. *Ind. Eng. Chem. Res.*, 33(9):2111, 1994.

V. S. Vassiliadis. *Computational Solution of Dynamic Optimization Problems with General Differential-Algebraic Constraints.* PhD thesis, University of London, 1993.

R. von Watzdorf, U. G. Näf, and C. C. Pantelides. Deterministic and stochastic simulation of batch/semicontinuous processes. *Computers Chem. Engng*, 18(S):S343–S347, 1994.

V. T. Voudouris and I. E. Grossmann. Mixed-integer linear programming reformulations for batch process design with discrete equipment sizes. *Ind. Eng. Chem. Res.*, 31(5):1315–1325, 1992.

V. T. Voudouris and I. E. Grossmann. Optimal synthesis of multiproduct batch plants with cyclic scheduling and inventory considerations. In *Proceedings AIChE Annual Meeting*, page Paper No. 135c, Miami, Florida, November 1992.

W. Rudin. *Principles of Mathematical Analysis.* McGraw-Hill, Inc., New York, 1976.

H. A. Watts. Algorithm 25: Starting step size for an ODE solver. *J. Comp. Appl. Math.*, 9:177–191, 1983.

H. S. Wellons and G. V. Reklaitis. Design of multiproduct batch plants under uncertainty with staged expansion. *Computers Chem. Engng.*, 13(1/2):115–126, 1989.

A. W. Westerberg, K. Abbot, and B. Allan. Plans for ASCEND IV: Our next generation equation-based modelling environment. In *AspenWorld 1994*, Boston, Massachusetts, November 6–9 1994.

J. H. Wilkinson. *Rounding Errors in Algebraic Processes.* Prentice-Hall, Inc., Englewood Cliffs, NJ, 1963.

J. A. Wilson. Dynamic model based optimisation in the design of a batch process involving simultaneous reaction and distillation. In *Institution of Chemical Engineers Symposium Series No. 100*, pages 163–181. Pergamon Press, 1987.

M. L. Winkel, L. C. Zullo, P. J. T. Verheijn, and C. C. Pantelides. Modelling and simulation of the operation of an industrial batch plant using gPROMS. *Computers Chem. Engng*, 19(S):S571–S576, 1995.

H. Wozniakowski. Numerical stability for solving nonlinear equations. *Numer. Math.*, 27:373–390, 1977.

Z. Xueya and R. W. H. Sargent. The optimal operation of mixed production facilities — a general fromulation and some approaches for the solution. In *The Fifth International Symposium on Process Systems Engineering*, pages 171–177, Kyongju, Korea, July 1994.

N. C. Yeh and G. V. Reklaitis. Synthesis and sizing of batch/semicontinous processes: Single product plants. *Computers Chem. Engng.*, 11(6):639–654, 1987.

S. E. Zitney and M. A. Stadtherr. Frontal algorithms for equation-based chemical process flowsheeting on vector and parallel computers. *Computers Chem. Engng*, 17(4):319–338, 1993.

S. E. Zitney, L. Brull, L. Lang, and R. Zeller. Plantwide dynamic simulation on supercomputers. *AIChE Symposium Series*, 91(304):313–316, 1995.

S. E. Zitney, J. Mallya, T. A. Davis, and M. A. Stadtherr. Multifrontal vs frontal techniques for chemical process simulation on supercomputer. *Computers Chem. Engng*, 20(6/7):641–646, 1996.

S. E. Zitney. Sparse matrix methods for chemical process separation calculations on supercomputers. In *Supercomputing '92*, pages 414–423, Los Alamitos, CA, November 1992. IEEE Comput. Soc. Press.