Solution Approaches to Large Scale Multistage Stochastic Programs with Endogenous and Exogenous Uncertainty

by

Brianna Christian

A dissertation submitted to the graduate faculty of
Auburn University
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Auburn, Alabama
December 16, 2017

Keywords: Optimization under Uncertainty, Multistage Stochastic Programming, Clinical Trial Planning, New Technology Investment Planning

Abstract

The focus of this dissertation is the investigation of approaches to address the computational challenges of solving multistage stochastic programs (MSSPs) with both endogenous and exogenous uncertainty. My work to date has considered two MSSP case study problems. The first problem is the pharmaceutical R&D pipeline management problem where the objective is to determine the clinical trial schedule which maximized expected net present value considering uncertainty in the outcome of each clinical trial. The second is a new technology evaluation problem where the objective is to determine an optimal investment strategy for new technology development given uncertainty in the cost of capacity expansion, yield, technology success, and product demand. The computational complexity of MSSPs increases as the size of the problem (i.e. number of uncertain parameters) increases. To address this complexity, this dissertation introduces three different approaches. In the first approach, the goal was to find a feasible solution for computationally intractable MSSPs. To accomplish this, two heuristic algorithms that generate feasible solutions for the pharmaceutical clinical trial planning problem were developed. The first heuristic solves a series of two-stage stochastic programs with shrinking planning horizon. The second one uses a knapsack problem based approach to select clinical trial investments. The knapsack problem-based heuristic was generalized to Expected Value Decomposition Algorithm (EVDA) and used to solve instances of the NTIP problem. The EVDA algorithm provided tight feasible bounds on the solution of the deterministic MSSP for the NTIP problem.

In the second approach, the computational complexity of the MSSPs is addressed using a branch and bound algorithm to find the true MSSP solution. The algorithm pairs the knapsack problem based decomposition heuristic with either an upper bound generated and updated using progressive hedging or optimal scenario solutions (OSS) to solve the clinical trial planning problem. The results reveal that both approaches reduce the computational resources required for solving large instances of the pharmaceutical clinical trial planning problem. The branch and bound algorithm with the OSS yielded a solution with a relative gap of 8% for a 7 product clinical trial planning problem which had previously been too large to generate and solve. The solution time for both algorithms makes the developed branch-and-bound algorithm unattractive for small problems, however, it can be used for problems which are too large to be generated using currently available computational resources.

The third approach to reduce the computational complexity of MSSPs is to reduce the number of constraints needed to ensure that the solution does not anticipate the outcome of the uncertain parameters. Often the MSSP is written where the scenarios included do not represent the full set of scenarios. As such, it can be difficult to solve the problem as the traditional approaches for constraint reduction do not necessarily apply. This dissertation introduces an algorithm for generating the minimum cardinality non-anticipativity constraint set for MSSPs where the scenario set is a subset of the full set of scenarios. The approach is generalizable for MSSPs with gradually or instantaneously realized, endogenous and exogenous uncertainty.

Acknowledgements

My deepest gratitude to my advisor, Dr. Selen Cremaschi, for her persistent guidance, patience, and constant encouragement throughout my doctoral program both at Auburn University and the University of Tulsa. Her dedication to my success both personally and professionally has been invaluable.

It is a pleasure to thank my dissertation committee, Dr. Jin Wang, Dr. Peter He, Dr. Mario Eden, and Dr. John Siirola. I am very grateful for the feedback provided in improving my dissertation. I would also like to thank my reader Dr. Lauren Beckingham who generously agreed to serve as my reader on short notice.

I would like to acknowledge funding from the US National Science Foundation through the Auburn University Integrative Graduate Research and Education Traineeship (IGERT) program (Award# 1069004), and the NSF Career Grant (Award # 1623417), as well as, computational resources provided by the Auburn University Hopper Cluster.

Table of Contents

List of Tables

List of Figures

CHAPTER 1
INTRODUCTION

Multistage stochastic programming (MSSP) is a scenario-based solution approach used to solve optimization problems under uncertainty. In chemical engineering, these problems arise in production planning and scheduling where the future demand is unknown, location selection and transportation management with uncertain transportation costs and market demand, and chemical product and process design where process performance is not known. Some example problems previously explored include the pharmaceutical clinical trial planning problem (Colvin and Maravelias, 2008), the oil field infrastructure planning problem (Gupta and Grossmann, 2011), the process synthesis problem (Goel & Grossmann, 2006), the open pit mining problem (Boland et al., 2008), and the vehicle routing problem (Khaligh and Mirhassani, 2015). The ability to solve multistage stochastic programming formulations is limited computationally by problem size. The growth in problem size is impacted by the number of uncertain parameters and the number of realizations for each uncertain parameter. As a result, most MSSPs solved in literature are limited to a few uncertain parameters with a few realizations for each parameter. These problems generally have between 8 and 1000 scenarios.

Often real-world size problems have many uncertain parameters with multiple realizations. The MSSPs of these problems quickly become computationally intractable. The goal of my work is to develop solution approaches which address the computational complexity of multistage stochastic programs. To address the computational complexity of MSSPs I have used two approaches. The first one investigates heuristic approaches which provide feasible solutions. The second approach seeks to address the computational complexity through the development of improved bounding and constraint reduction procedures.

1.1 Objectives

- Exploration of heuristic approaches to generate approximate solutions for MSSPs
- Identification of approaches to address the computational complexity of solving large-scale MSSPs
- Investigation of approaches for non-anticipativity constraint generation in in MSSPs with incomplete scenarios sets and gradually realized endogenous and exogenous uncertainty

1.2 Organization

Chapter 2 introduces the theoretical background which includes a section describing different approaches used for solving optimization problems where there is uncertainty in parameter values (Section 2.1). Chapter 2 also describes the stochastic programming framework (Section 2.2) and covers the relevant literature describing the application and advancements in multistage stochastic programming (Section 2.3). In Chapter 3, two different motivating problems and their multistage stochastic programming formulations are introduced. Section 3.1 introduces the pharmaceutical clinical trial planning problem and Section 3.2 describes a new technology investment planning problem. The proposed heuristic algorithms are introduced in Chapters 4 and 5. Chapter 4 introduces a multiple two-stage decomposition algorithm (Section 4.1). The algorithm was applied to the pharmaceutical clinical trial planning problem, and the results are summarized and discussed in Section 4.2. In Chapter 5, a knapsack decomposition algorithm (KDA) is introduced (Section 5.1). Section 5.2 presents and discusses the results of solving pharmaceutical clinical trial planning problem using the knapsack decomposition algorithm. Variations to the heuristic rules of the KDA are introduced in Section 5.4. Descriptions of the case studies used to test the variations are presented in Section 5.5. Impacts of these algorithm variations on the solution time and the objective function value of these computational experiments are compiled in Section 5.6. The KDA is generalized to accommodate continuous decision variables and non-trivial recourse actions in Section 5.7. A branch and bound algorithm to solve large scale MSSPs is presented in Chapter 6, where Section 6.1 introduces the branch and bound algorithm, and Section 6.2 presents two approaches for generating dual bounds for the MSSP. The first approach employs a progressive hedging algorithm. The second uses Optimal Scenario Solutions (OSS). In Section 6.3, a variation of the KDA algorithm is presented. The variation provides a primal bound for MSSPs. Section 6.4 presents a set of pharmaceutical clinical trial planning case studies. The results and discussion of the case studies is covered in Section 6.5. Conclusions on the branch and bound algorithm are summarized in Section 6.6. Chapter 7 presents an algorithm for generation of the minimum cardinality set of non-anticipativity constraints. The algorithm can be used to generate constraint sets for MSSPs with either gradually or instantaneously realized endogenous or exogenous uncertainty when the scenario set is a subset of the full set. Background for the algorithm is presented in Section 7.1. The algorithm itself is given in Section 7.2. Case studies with varying number of uncertain parameters and number of realizations for each uncertain parameter, and the results of applying the algorithm to generate NACs for these case studies are discussed in Section 7.3, followed by conclusion in Section 7.4. Overall conclusions and future directions are summarized in Chapter 8.

CHAPTER 2
BACKGROUND

2.1 Solving Optimization Problems with Uncertainty

Simulation optimization (SIMOPT), dynamic programming, robust optimization, and stochastic programming are used to solve optimization problems under uncertainty in literature. Dynamic programming, robust optimization, and stochastic programming can be classified as conventional optimization approaches where the uncertainty is explicitly incorporated in the problem formulation. On the other hand, SIMOPT is a non-conventional optimization approach which combines discrete event simulation and combinatorial optimization frameworks (Pekny, 2002). Algorithms utilizing a SIMOPT strategy often use a two-loop process where the inner loop generates and solves a set deterministic optimization problems where the parameters of each problem correspond to unique realizations of the uncertain parameters. The outer loop then gathers the information found in the inner loop and uses the information to perform risk analysis or update parameters in the inner loop (Blau et al., 2004).

Dynamic programming is a mathematical programming technique used to handle multistage decision processes. In dynamic programming, uncertainty is considered an inherent part of the multistage decision process (Bellman, 1954). Uncertainty is handled by decomposing the problem into series of sub-problems (decision stages) over time. For each decision stage, there is a value associated with each state of the problem. The state of the problem, in this case, refers to the knowledge of the values of uncertain parameters at the current decision stage. Values at each decision stage are calculated recursively. The effectiveness of dynamic programming relies on the problem having an optimal substructure and overlapping sub-problems. An optimal substructure refers to a problem having a globally optimal solution which can be constructed with locally optimal sub-problems. The problem is said to have overlapping sub-problems if the problem can be broken down into a sub-problem where the structure of the sub-problem is identical for all sub-problems. Solving dynamic programming problems is computationally intensive because it suffers from the curse of dimensionality.

Robust optimization is an approach for handling stochastic optimization problems. The approach uses a deterministic optimization framework coupled with a set-based approach to handle the parameter uncertainty. (Brown and Caramanis, 2011) The first step in solving the robust problem is to formulate the robust counterpart problem. (Ben-Tal et al., 2009) The robust counterpart problem is formed by incorporating the uncertainty set directly into deterministic formulation. Solving the robust counterpart problem results in the optimal solution which satisfies all realizations of uncertainty. Often considering the full set of uncertainty results in a worst-case solution. Recent work in robust optimization has looked at approaches to improve the quality of the solution by limiting or modifying the uncertainty set.

Stochastic programming (SP) frameworks rely on the knowledge of the distribution or the ability to estimate the distribution of the uncertain parameters. (Birge & Louveaux, 2011) Unlike robust optimization, SP uses the knowledge of the distribution of the uncertain parameters to find a solution that is feasible for all considered realizations (scenarios). Uncertainty is explicitly accounted for by incorporating the realizations of uncertainty directly into the objective function. Uncertainty is accounted for using a statistical measure. Often a statistic such as expected value (EV) or conditional value at risk (CVAR) is used. Of the conventional approaches discussed, the stochastic programming approach is easiest to formulate. Stochastic programming is not limited to uncertainty in either the objective function or in the constraints. Incorporation of uncertainty through expected value allows for a mathematical measure of tradeoff between robustness and performance which is not the case in the robust optimization formulations. Sahinidis (2004) presents a thorough review of each of the conventional approaches discussed.

2.2 The Stochastic Programming Framework

The stochastic programming framework uses a scenario based approach. The major components in a multistage stochastic program are (1) the set of scenarios which represent the set of all possible outcomes of uncertainty, and (2) a set of stages where decisions can be made and recourse actions can be taken (Birge and Louveaux, 2011). Scenarios in SPs are generated using the possible realizations of the uncertain parameters. When all uncertain parameters can be considered independent, the scenario set can be generated using the Cartesian product of the outcomes of each uncertain parameter (Apap and Grossmann, 2015). Suppose that the outcomes of an uncertain parameter, μ_k, can be represented using a probability density function p_k. Generally, the parameter distributions are either assumed to be discrete or approximated with a discrete distribution,f_k, with n possible realizations ($\Omega_k = \omega_1, \omega_2, \ldots, \omega_n$). Discretizing the probability density function also discretizes the scenario set. This yields a finite number of scenarios. The full set of scenarios for $|\boldsymbol{K}|$ independent uncertain parameters is then constructed using the Cartesian product of the realizations of each uncertain parameter ($\Omega_1 \times \Omega_2 \times \ldots \times \Omega_k$). In cases where the uncertain parameters are not independent, scenarios may or may not accurately be represented using the Cartesian product.

Assuming each uncertain parameter is independent, the probability of a scenario occurring can be obtained as, $P(\Omega_1 \cup \Omega_2 \cup \ldots \cup \Omega_k) = \prod_{k \in \boldsymbol{K}} P(\Omega_k = \omega)$ (Ross, 2006a). If independence assumption is not valid, the probability of a scenario occurring is calculated using conditional probabilities. (Ross, 2006b)

The second component of SPs is stages. Stages represent points where uncertainty is realized and new decisions can be made. In SPs, stages are typically tied to time periods, meaning that decisions will be made at one time period, and uncertainty realizations and new decisions will occur at a later time period. Two-stage SPs have one decision stage, and one stage where uncertainty realizations occur. When realizations occur, scenario specific recourse actions can be taken. Multistage stochastic programs have multiple stages where decisions are made and multiple stages where uncertainty is realized and recourse actions are taken (Birge and Louveaux, 2011). Figure 2.1 illustrates the relationship between the decision stages and the realization stages in MSSPs.

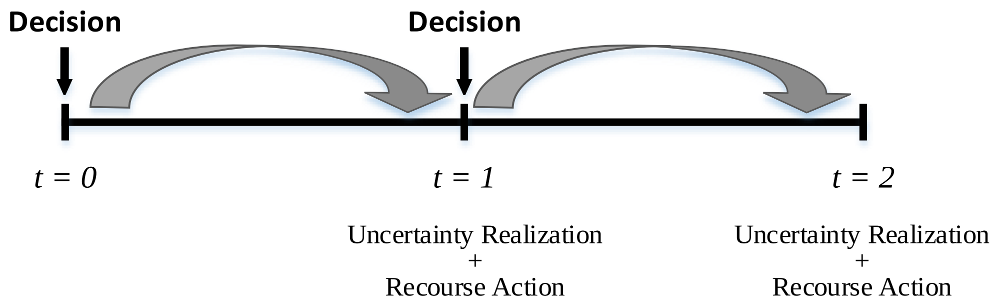

Decision
Decision
t = 0
t = 1
t = 2
Uncertainty Realization + Recourse Action
Uncertainty Realization + Recourse Action

Figure 2.1 Relation of decision stages, uncertainty realization, and recourse action stages in MSSPs

The uncertainty in SPs can be grouped into two broad categories, endogenous and exogenous. The realization of exogenous uncertain parameters is not affected by the problem decisions. (Jonsbraten et al., 1998) For instance, demand is usually considered an exogenous uncertain parameter. In contrast, values of endogenous uncertain parameters are impacted by decisions. The impact can either be in the resolution or in the distribution of the uncertain parameter. For example, when new product development is considered, the realization of a product completing a stage successfully is not revealed until that development stage is completed, and to complete that stage, the decision to start that development stage should be made. This type of uncertainty is called Type-2 endogenous uncertainty. The second type of endogenous uncertainty considers uncertainty in the distribution of the uncertain parameters. An example of this type of uncertainty would be facility protection problem where the likelihood of a facility failing to deliver goods or services after a disruptive event depends on the level of resources allocated as protection to that facility (Medal et al., 2016).

At the beginning of the planning horizon and before any decisions are made, all scenarios are indistinguishable because no uncertainty has been realized. Therefore, the initial decisions must be identical for every scenario. As uncertainty is revealed, either through decisions or naturally, the scenarios begin to be distinguishable. Once a scenario or set of scenarios is differentiable from the other scenarios, decisions for that scenario or set of scenarios may be made independently. This concept called non-anticipativity can either be enforced implicitly through variable definition or through the use of logical constraints, called non-anticipativity constraints (NACs). In SPs, NACs ensure that decisions for indistinguishable scenarios do not anticipate the realization of the uncertain parameter.

The ability to solve stochastic programming formulations are limited computationally by problem size. The problem grows exponentially with a linear increase in number of uncertain parameters. The exponential growth is caused by the growth in the number of scenarios. When considering independent realizations of uncertain parameters, the scenario set is defined by the Cartesian product. For n uncertain parameters with k realizations, the result of the Cartesian product is a set of k^n ordered pairs (scenarios). The number of scenarios affects the size of variables and the number of constraints in a stochastic program. Increasing the number of scenarios requires the addition of new NAC to ensure that a feasible solution is produced. Adding NACs directly affects the solution time of the problem.

Most literature to date has studied strictly exogenous uncertainty. In this work, we are specifically interested in solving problems with both endogenous and exogenous uncertainty. Literature considering both endogenous and exogenous uncertainty is extremely limited. As such, the literature review discusses works considering endogenous uncertainty and works with both endogenous and exogenous uncertainty.

2.3 Literature on Solving Multistage Stochastic Programs with Endogenous Uncertainty

Jonsbraten et al. (1998) was one of the first to study optimization under endogenous uncertainty. The authors developed a branch-and-bound algorithm coupled with complete enumeration to generate scenario trees. Their approach was designed to tackle problems where only the first-stage decisions were impacted by the endogenous uncertainty and the endogenous uncertain variables were represented by Bernoulli distributions. They solved several instances of sub-contracting problems which have endogenous uncertainty in the in-house production costs of the parts. Their results illustrated the impact of decision-dependent uncertainty on the optimal decisions.

Goel and Grossmann (2004) considered an approximation approach, which searched a sub-space of the feasible region to find a "good" solution. They used this approach for solving an offshore oil infrastructure planning problem. The goal was to determine the optimal placement of production platforms in order to minimize the risk of obtaining a negative net present value. The sub-space is obtained by removing the scenario dependency of investment decisions for uncertain fields, yielding a more constrained version of the original problem. Then, this constrained version is solved to optimality. The results revealed that the solutions obtained to the constrained formulation were significant improvements compared to the deterministic solutions. Goel et al. (2006) presented a branch-and-bound algorithm that generates upper-bounds using the solution of the Lagrangean dual problem with relaxed NACs. The lower bounds were generated heuristically based on the solution of the Lagrangean dual problem. The results suggested that the branch-and-bound algorithm achieved significantly better solutions and tighter optimality gaps than the heuristic presented in Goel and Grossmann (2004).

Tarhan and Grosmann (2008) presented a MSSP for solving process synthesis problems with decision-dependent uncertainty. Uncertainty was assumed to be in the yields of process equipment and was revealed gradually as investments in process equipment occurred. The authors proposed a solution strategy that used a duality based branch-and-bound algorithm. The strategy was able to solve a 16 scenario example to within 3% optimality. Tarhan et al. (2009) explored an offshore oil planning problem with endogenous uncertainty in the initial maximum oil flowrate. Extending the work of Goel and Grossmann (2004, 2006), Tarhan et al. (2009) considered a single oil field with non-linear reservoir model and a gradual realization of uncertainty. Using a duality-based branch-and-bound, the authors located solutions that were up to 22% better than solutions obtained using an expected value approach. They, however, noted that the solution times for the MSSP were “rather long”.

Mercier (2009) proposed a multi-step anticipatory algorithm, which uses a sample average approximation approach to generate a Markov Decision Process (MDP). The MDP is then solved, and the greedy solution is returned. The algorithm was tested using 12 instances of the pharmaceutical R&D pipeline management problem. Solution qualities for the algorithm were 10% better than the dynamic programming equivalent for all instances. The authors concluded that the algorithm was, nevertheless, computationally expensive when applied to the pharmaceutical R&D pipeline management problems.

Colvin and Maravelias (2009) explored a rolling-horizon approximation approach, which yielded tight feasible solutions for large instances of the pharmaceutical clinical trial planning problem. The authors divided the planning horizon into a finite number of subsets. A relaxed MSSP is generated for the first subset by removing all inequality non-anticipativity constraints (NACs) for the stages beyond the first subset. The solution of the relaxed MSSP is implemented for the first subset, and related uncertainty is realized. Then, the process is repeated for each subset until the end of the planning horizon is reached. The authors were able to successfully solve cases with more than 1000 scenarios. Another approach, a branch-and-cut algorithm (Colvin and Maravelias, 2010), initially adds a percentage of NACs and then iteratively adds constraints based on violations. The authors concluded that the algorithm reduces the number of NACs that should be included in the problem formulation significantly and that this method would be advantageous for any problem where the majority of constraints are NACs.

Solak et al. (2010) proposed a sample average approximation (SAA) based algorithm to generate candidate solutions for the R&D project portfolio optimization problem. The presented approach generates M sample subsets of scenarios and M

number of MSSPs for each scenario subset. For each of the smaller MSSPs, the upper-bound is obtained via solving its Lagrangean dual using a modified sub-gradient algorithm to update the Langrange multipliers. The lower bound is generated via a heuristic that searches for a feasible solution in the vicinity of the Lagrangean dual solutions obtained during each iteration of the sub-gradient algorithm. The authors suggest that a branch-and-bound algorithm may be used to close the duality gap if necessary. The solutions of the M MSSPs are called candidate solutions. The quality of the first-stage decisions of the candidate solutions are evaluated using a larger sample set of the scenarios. By repeating the process, the variance of the results is used to find a bound on the true solution. Computational studies included two technology portfolio examples where five and 10 projects were considered. The results revealed that the algorithm was able to generate solutions with an estimated optimality gap of around two percent.

Tarhan et al. (2013) presented an approach that improves upper bounds for solving non-convex MINLPs with decision-dependent uncertainty. The algorithm solves the Lagrangean relaxation of the dual problem where NACs have been removed to obtain the upper bound. The lower bound is generated by locating a feasible solution using a rolling-horizon approach. The authors solved two non-convex non-linear problems. The first problem is a version of the process synthesis problem presented in Goel and Grossmann (2006). The second is an offshore oil field planning problem originally presented in Tarhan et al. (2008). The authors concluded that using the outer approximation solution to upper bound the intermediate problems generated during the branch-and-bound algorithm rather than solving them to optimality reduced the solution time of both problems by 60%, and the solutions of the intermediate problems remained within 0.01% of their optimal solutions.

More recently, Gupta and Grossmann (2014) proposed a Lagrangean decomposition scheme that utilizes a scenario grouping strategy, which allows partial decomposition of the full space model. Scenarios are grouped based on differences in one endogenous parameter. The strategy was applied to the oilfield planning problem from Goel and Grossmann (2004) and the process network problem originally presented in Goel and Grossmann (2006). The authors concluded that their scenario grouping strategy yielded tight lower bounds on medium sized problems; however, further work is needed to test its effectiveness on larger problems.

Apap and Grossmann (2015) developed a sequential scenario decomposition heuristic. The heuristic solves a series of endogenous sub-problems to determine binary investment decisions, fixes the binary investment decisions to satisfy the first-period endogenous and exogenous non-anticipativity constraints, and solves the resulting model to obtain a feasible solution. The author used their heuristic in conjunction with Lagrangean relaxation strategy (lower bound) to solve the fullspace problem with endogenous and exogenous uncertainty. To test their algorithm, the authors considered process synthesis problem (Goel and Grossmann ,2006) and the offshore oil planning problem (Gupta and Grossmann, 2014). The authors concluded that the heuristic found a high-quality feasible solution several orders of magnitude quicker than solving the full space model.

A preprocessing step for NAC reduction in the vehicle routing problem was presented in Khaligh and Mirhassani (2015). Non-anticipativity constraint reduction was achieved by removing constraints that would never be active due to the problem formulation. The authors tested their preprocessing step using several instances of the vehicle routing problem. The results showed that the preprocessing step reduced the necessary computation time in each instance. The authors concluded that the develop NAC preprocessing step did not impact the quality of the lower-bound of the problem.

CHAPTER 3

MULTISTAGE STOCHASTIC PROGRAMS FOR TWO PLANNING PROBLEMS

3.1 The Pharmaceutical R&D Pipeline Management Problem

Motivation for frameworks which guide decision-making for new-product-portfolio management are being prompted by rising attrition (failure to succeed) rates for new drug products. (PwC, 2012) Rates of attrition in the Phase II-proof of concept and Phase III clinical trials for the top 100 companies have increased by 10% since 1990. (PwC, 2012) The rise in these rates is attributed to poor management of candidate products. Often the decision of which products to invest in are determined by researchers, as a results candidate products which showed only marginal efficacy are frequently selected for investment. As such, it is important to develop a framework which balances the potential value of a candidate product with the risk of failure. Mathematical approaches provide a flexible non-biased way of representing new-product portfolios.

The clinical trial planning portion of the pharmaceutical R&D pipeline management problem can be characterized by a set of new product development projects. Each candidate product is required to complete a series of ordered clinical trials before reaching the market. Investments in the development of candidate products is limited based on resource availability. Whether a product successfully completes each trial, the trial duration, and the total required resources of each trail are not necessarily known a priori. The goal of the problem is to determine the clinical trial schedule which results in the maximum expected return. Solutions to the mathematical program have been obtained for problems with as many as seven candidate products (Colvin & Maravelias, 2010), however, the solution approaches have failed to solve real-world sized problems which can have up to 500 candidate products. (PwC, 2012).

Each fixed duration clinical trial has a known monetary and resource(s) cost. Decisions as to when to start each trial are made along a discrete n-month planning horizon divided into evenly spaced time steps. The number of investment decisions are limited by resource availability. Once a new product successfully completes all clinical trials, revenue associated with achieving market availability is realized. It is assumed that the revenue for reaching market availability is known with certainty. Penalties are accessed for delay of active patent life and reduced market share. The objective of the problem is to determine the clinical trial schedule which maximizes the expected net present value (NPV) given that the outcome of each clinical trial is not known with certainty. The problem is constrained to ensure that (1) each clinical trial is only performed once, (2) to ensure that clinical trials are performed in the correct order, and (3) the utilized resources at any given time period do not exceed the available resources. Uncertainty in clinical trial outcome can be visualized in Fig. 3.1.

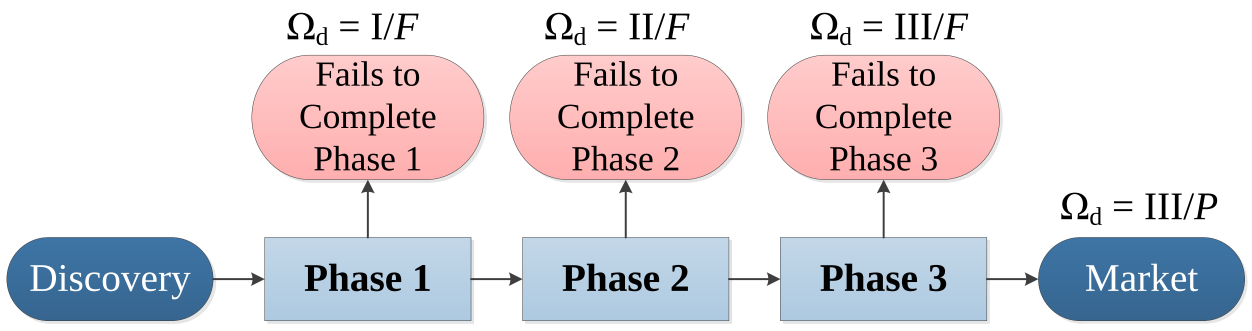


Figure 3.1 Representation of uncertainty in the clinical trial planning problem

Uncertainty in the clinical trial planning problem is assumed to only exist in the outcome of each clinical trial. For simplicity, it is also assumed that each clinical trial is a Bernoulli trial with a known probability of success. For each product, there is a set of three uncertain parameters, $\xi_{d,j}$, which can be viewed as discrete random variables. The uncertain parameters refer to the outcomes of each clinical trial. Outcomes for each clinical trial can either be pass (P) or fail (F). For each product, we can enumerate all possible outcomes. Because a clinical trial will not be carried out if the predecessor trial is a failure, we can reduce the number of outcomes to $|\mathbf{J}|+1$. In the case of three clinical trials, $\mathbf{J}$ = {Phase I, Phase II, Phase III}, the possible outcomes of each product are given as $\Omega_d = \{\text{I}/F, \text{II}/F, \text{III}/F, \text{III}/P\}$. In set Ω_d, the first term of each element represents the trial, and the second one represents the outcome. For example, the element I/*F* corresponds to the outcome of a product's failure of the first trial. The probability of each outcome is calculated assuming that the trials are a set of independent Bernoulli trials. For instance, if a drug has a 30% chance of passing trial one, a 50% chance of passing trial two, and an 80% chance of passing trial three, $\mathbf{P}(\Omega_d=\text{II}/F)$ would be expressed as $\mathbf{P}(\xi_{d,1}=\text{Pass})\mathbf{P}(\xi_{d,2}=\text{Fail}) = 0.3\ (1 - 0.5) = 0.15$. Scenarios are created using combinations of possible outcomes for each product. This notation reduces the number of scenarios from $2^{|\mathbf{J}||\mathbf{D}|}$ to $|\mathbf{J}+1|^{|\mathbf{D}|}$. Assuming the outcome of each product is independent, the probability for realizing each scenario is expressed as $\mathbf{P}(\Omega_1 \cap \Omega_2 \cap \ldots \Omega_D) = \prod \mathbf{P}(\Omega_i)$. For a detailed discussion of outcomes and scenario generation, we refer the reader to Colvin and Maravelias (2008). Non-anticipativity constraints are added to ensure that all decisions regarding clinical trials are identical until the differentiating trial is completed.

The objective is to maximize the expected net present value (ENPV) of the decision tree. The net present value of each scenario is calculated by deducting the cost(s) of the decisions and the penalties from the total realizable revenue. The realizable revenue includes the depreciated revenue of the products that successfully completed all trials and the depreciated potential revenue of products whose trials were continuing or were not started within the planning horizon. The rigorous MSSP of the problem described in this section was originally developed by Colvin and Maravelias (2008), and is included in Appendix A.

3.1.1 Clinical Trial Planning Case Studies

In this section, we will define six clinical trial planning case study problems. The case study problems are used throughout this dissertation as benchmark problems for testing algorithms. Case studies have between two and six candidate products. The parameters for each case study are shown in Tables 3.1-3.6

Table 3.1 Parameters of the two-drug case study

	Duration (time periods)		**Probability of Success**		**Cost($1M)**		**Resource 1 (max =2)**		**Resource 2 (max=3)**		rev^{max}	γ^L	γ^D
Drug	PI	PII	PI	PII	PI	PII	PI	PII	PI	PII			
D1	2	4	0.3	0.5	10	90	1	1	1	2	3100	19.2	44
D2	2	3	0.4	0.6	10	80	1	2	1	1	3250	19.6	56

** Clinical trial plan for a 15-month planning horizon divided into five equal time period

Table 3.2 Parameters for a two-drug, three-trial case study

	Duration (time periods)			**Probability of Success**			**Trial Cost ($M)**			**Resource 1 (max =2)**			**Resource 1 (max =3)**			rev^{max}	γ^L	γ^D
Drug	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII			
D1	2	4	4	0.3	0.5	0.8	10	90	220	1	1	2	1	2	3	3100	19.2	22
D2	2	3	5	0.4	0.6	0.8	10	80	200	1	2	2	1	1	3	3250	19.6	28

** Clinical trial plan for a 15-month planning horizon divided into five equal time periods

Table 3.3 Parameters for a three-drug case study

	Duration (time periods)			**Probability of Success**			**Trial Cost ($M)**			**Resource 1 (max =2)**			**Resource 1 (max =3)**			rev^{max}	γ^L	γ^D
Drug	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII			
D1	2	4	4	0.3	0.5	0.8	10	90	220	1	1	2	1	2	3	3100	19.2	22
D2	2	3	5	0.4	0.6	0.8	10	80	200	1	2	2	1	1	3	3250	19.6	28
D3	2	3	4	0.3	0.6	0.9	10	90	180	1	1	2	1	1	3	3300	20	26

** Clinical trial plan for a 36-month planning horizon divided into 12 equal time periods

Table 3.4 Parameters for a four-drug case study

	Duration (time periods)			**Probability of Success**			**Trial Cost ($M)**			**Resource 1 (max =4)**			**Resource 2 (max =3)**			rev^{max}	γ^L	γ^D
Drug	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII			
D1	1	1	3	0.3	0.5	0.8	10	90	220	1	1	2	1	2	3	3100	19.2	22
D2	1	2	2	0.4	0.6	0.8	10	80	200	1	2	2	1	1	3	3250	19.6	28
D3	1	1	3	0.3	0.6	0.9	10	90	180	1	1	2	1	1	3	3300	20	26
D4	1	2	2	0.4	0.6	0.8	10	100	170	1	1	2	1	2	3	3000	19.4	24

** Clinical trial plan for a 36-month planning horizon divided into six equal time periods

Table 3.5 Parameters for a five-drug case study

	Duration (time periods)			Probability of Success			Trial Cost ($M)			Resource 1 (max =4)			Resource 2 (max =3)			rev^{max}	γ^L	γ^D
Drug	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII			
D1	1	1	3	0.3	0.5	0.8	10	90	220	1	1	2	1	2	3	3100	19.2	22
D2	1	2	2	0.4	0.6	0.8	10	80	200	1	2	2	1	1	3	3250	19.6	28
D3	1	1	3	0.3	0.6	0.9	10	90	180	1	1	2	1	1	3	3300	20	26
D4	1	2	2	0.4	0.6	0.8	10	100	170	1	1	2	1	2	3	3000	19.4	24
D5	1	2	3	0.35	0.5	0.9	10	70	210	1	1	2	1	1	3	3150	19.6	24

** Clinical trial plan for a 36-month planning horizon divided into six equal time periods

Table 3.6 Parameters for a six-drug case study

	Duration (time periods)			Probability of Success			Trial Cost ($M)			Resource 1 (max =4)			Resource 2 (max =3)			rev^{max}	γ^L	γ^D
Drug	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII			
D1	1	1	3	0.3	0.5	0.8	10	90	220	1	1	2	1	2	3	3100	19.2	22
D2	1	2	2	0.4	0.6	0.8	10	80	200	1	2	2	1	1	3	3250	19.6	28
D3	1	1	3	0.3	0.6	0.9	10	90	180	1	1	2	1	1	3	3300	20	26
D4	1	2	2	0.4	0.6	0.8	10	100	170	1	1	2	1	2	3	3000	19.4	24
D5	1	2	3	0.35	0.5	0.9	10	70	210	1	1	2	1	1	3	3150	19.6	24
D6	1	2	3	0.45	0.45	0.8	10	85	195	1	2	2	2	1	3	3050	19	25

** Clinical trial plan for a 36-month planning horizon divided into six equal time periods

For each case study we were able to generate and solve the deterministic equivalent MSSP using the formulation presented in Colvin and Maravelias (2008). The objective function values, solution times, and optimality gaps for each problem are shown in Table 3.7.

Table 3.7 Objective function values, optimality gaps, and solution times for the six clinical trial benchmark problems

	MSSP Objective Function Value	Optimality Gap	MSSP Solution Time (CPU Second)
Two-Product Two-Trial	$1104 M	0.1%	0.09
Two-Product Three-Trial	$800 M	0.1%	0.22
Three-Product	$1189 M	0.1%	2.85
Four Product	$1697 M	0.1%	8.00
Five-Product	$2083 M	0.1%	44.90
Six-Product	$2407 M	3.0%	253.64

3.2 The New Technology Investment Planning Problem (NTIP)

An expanding middle class, urbanization, and pushes for sustainability, have driven growth in chemical demand. (ExxonMobil, 2015) Demand for chemicals is expected to grow by 45% over the next 10 years (ExxonMobil, 2015). ExxonMobil is "progressing strategic investments that will capture low-cost feedstocks and increase premium product capacity to supply growing markets". DuPont plans to continue to drive innovation and accelerate growth in developing markets (DuPont, 2015). Bayer states that "being successful as a Life Science company requires a pronounced innovation culture that is the breeding ground for new ideas and facilitates their translation into successful products" (Bayer, 2015). These trends suggest that the feedstock and product portfolios along with utilized technologies of the chemical process industry (CPI) may grow and be quite different compared to today's in the near future. As such, there is tremendous opportunity for investigating the impacts of these new technologies/feedstocks/products on the chemical process industry.

Determining which feedstock or technology to develop is a challenging task. Often there are many emerging feedstocks and processing technology alternatives. Incorporating new feedstocks and technologies into the chemical process industry (CPI) is an expensive and time consuming process. Beyond initial discovery, investments into new technology serve two purposes: (1) to improve the efficiency of the process, and (2) to expand the production capacity of the technology to meet market demands. Generally, the performance of a technology is not known with certainty until after the technology is fully developed. Large investments into technology projects do not guarantee successful development or favorable results. Often new technology projects are mishandled or mismanaged leading to investments without a return or loss of investment due to project abandonment (Cooper, 2007). Furthermore, the development of technologies is partially driven by product demand, which is not known with certainty at the time of the investments. Therefore, to incorporate new feedstocks and technologies into the existing CPI infrastructure, a systematic approach is necessary to answer the following questions: how much to invest, which new technologies to invest in, and when to invest in each technology for research and development (R&D) and for capacity expansions?

New chemical technologies start as ideas. Investments in these ideas lead to laboratory experimentation and if successful, pilot plant and commercial installations. Determining when and how much to invest in a new technology is a complicated decision, which relies on knowledge of the cost, yield, and the success of the new technology, as well as, the continued demand for the product that the technology produces. Figure 3.2 provides a visual for the development and incorporation of new technologies into the CPI. The mini CPI depicted in Fig. 3.2 contains three chemicals (CHEM1, CHEM2, and CHEM3) and two technologies (TECH1 and TECH2). Each chemical can be purchased, produced (arrows connecting to technologies), and/or sold to meet demand. Technologies connected with solid lines represent established production routes. Connections between chemicals and technologies which appear as dashed lines are considered potential production routes.

They are defined as technologies which can be used to produce its output chemical(s) but have not yet been developed enough for commercial production. The goal of new technology investment planning (NTIP) problem is to determine when and how much to investment in the development of potential production routes in order to minimize the total cost of production over a planning horizon.

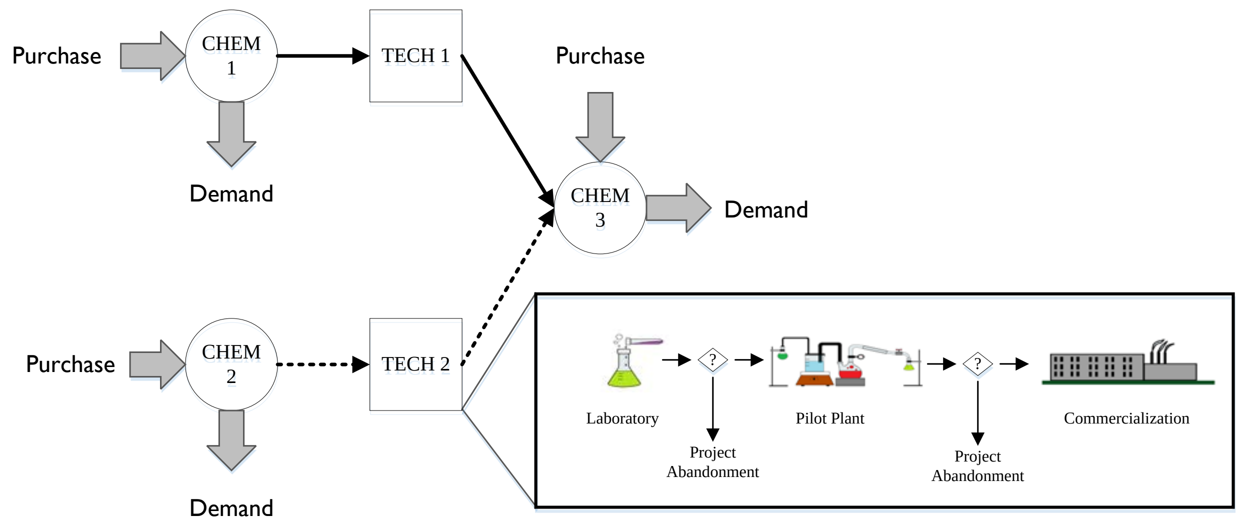


Figure 3.2 The NTIP Problem

The NTIP problem contains components of three well-established problems: production planning (Mula et al., 2006), capacity planning (Martínez-Costa et al., 2014), and R&D pipeline management (Verderame et al., 2010). The first problem deals with determining which route to use for the production of each chemical, and the second one determining how much capacity to install for each technology to produce each chemical. Technology development portion of the NTIP problem can be treated as a special case of the R&D pipeline management problem.

The remainder of this section introduces a multistage stochastic programming (MSSP) formulation that determines an optimal investment decision strategy for the NTIP problem considering uncertainties in the costs and future performances of new technology alternatives, and in the demands of the products. In Section 3.2.1, we present a literature review, which discusses different modelling and solution approaches that combines production and capacity planning. Section 3.2.1 also reviews relevant literature on R&D pipeline management under uncertainty. The NTIP model incorporates models for the cost of capacity expansion, the network representation of the CPI, and the R&D pipeline management. The forms and limitations of these models are discussed in Section 3.2.2. The MSSP formulation of the NTIP problem is presented in Section 3.2.3. The deterministic equivalent formulation of the MSSP is a large-scale non-convex mixed integer nonlinear program (MINLP), and cannot be solved to optimality due to its complexity. Here, we develop a linear approximation of the MINLP, and solve this model to bound the MINLP and provide an approximate solution. To construct the linear approximate model, the non-linear terms of the MINLP are replaced using linearly-segmented tight relaxations (Misener et al., 2011) and are linearized using exact linearization (Muhittin and Ossama, 1992) as appropriate. The linear approximate model is presented in Section 3.2.4. In Section 3.2.5, we solve the linear approximate model to determine the investment plan for four case studies. The first case study considers whether to replace production from one technology with a new undeveloped technology. The second case study involves simultaneous development of two competing technologies for a product that is new to the market with no existing technology to produce it. A third case study evaluates the development and penetration of a new technology for an existing product. The final case study considers a biomass to commodity chemical system, which evaluates investment decisions for the development of technologies for conversion of biomass to ethylene. Conclusions and future directions are summarized in Section 3.2.6.

3.2.1 Literature Review

3.2.1.1 The Production and Capacity Planning Problems

The NTIP includes aspects of both capacity and production planning problems. A review of literature considering uncertainty in the capacity and production planning problems can be found in Mula et al. (2006). In this section, we limit our discussion to works simultaneously considering capacity and production planning under uncertainty. Often the objective of the problem is to minimize the cost of production given uncertainty in either supply or demand, which can either be associated with the internal production route (i.e., the yield of a production technology) or can be external (i.e., the market demand of the product) (Kempf et al., 2011).

Gardner and Buzacott (1999) addressed the uncertainty of technology development for a direct steel making process for the production of steel plates. Decisions included the capacity utilization of alternative production routes, levels of capacity expansion, and amount of commodity to purchase. The authors concluded that their approach found a compromise solution bridging over 60 percent of the gap between the highest expected value and perfect information scenario. Goyal and Netessine (2007) used a game-theory approach to determine capacity and production decisions. The authors considered which technologies to use, how much capacity each technology should have, and the production route. The model included uncertainty in the demand of the final product based on competing firms. Dal-Mas et al. (2011) developed a formulation for strategic capacity investment decisions in the conversion of biomass to ethanol. The authors considered uncertainty in market prices. They used a case study optimizing the strategic decisions for the conversion of biomass to ethanol in northern Italy.

One modelling approach used by several authors is a real-options approach. Dangl (1999) solved the capacity planning problem using a real options approach combining optimal investment timing with optimal capacity choices. Demand was considered uncertain. Hagspiel et al. (2016) solved for optimal investment strategy under demand uncertainty using a real options approach. Decisions included timing of investments, quantity of production, and level of capacity. The authors considered two cases, the first was a flexible production case where the amount produced could be adjusted to meet demand. The second case considered an inflexible scenario where production levels were fixed at capacity level. The authors hypothesized that high uncertainty and high flexibility would result in later and larger investments.

Several other authors have considered stochastic programming solutions to the capacity and production planning problem. Eppen et al. (1989) developed a stochastic programming model for capacity planning of automobile production with the goal of maximizing profits. Decisions in the model were to determine automobile production levels at different locations given uncertainty in the future demand. In their problem, each production location could be retooled in order to produce a different set of products. Chen et al. (2002) considered a multi-product fixed and flexible capacity expansion problem with stochastic demand motivated by pharmaceutical production. The authors used a SP approach to solve problems with up to 5000 scenarios. Kostin et al. (2011) considered an integrated capacity and supply chain model under demand uncertainty. The authors modelled the process of sugarcane conversion to bioethanol including transportation and facility capacity expansion. The model was formulated using a two-stage SP approach where design decisions such as capacity expansion plans and the number of production and transportation units were made in the first stage. In the second stage, "wait-and-see" decisions such as production levels were determined.

3.2.1.2 The R&D Pipeline Management Problem

The second portion of the NTIP problem considers new technology development under uncertainty. The new technology development is an instance of the general new product/process development problem (Cooper, 2007). We define the new product/process development problem using a set of potential new development projects, a set of required tasks, and a set of limiting resources. The success of each project is conditional on successfully completing each of required tasks. Execution of tasks is limited based on the availability of a set of resources. The objective of the problem is to determine the schedule of tasks, which provides the maximum returns. In the NTIP problem, each new technology can be considered a new development project where the stages of development (i.e. laboratory, pilot plant, etc.) represent the tasks that need to be completed.

The general new product/process development problem has been studied extensively in literature. Verdereme et al. (2010) provides a comprehensive review of relevant literature on the general new product/process development problem under uncertainty. The problem has been modeled and solved using several approaches. The main approaches seen in literature are, simulation-optimization (Blau et al, 2004; Blau et al., 2000; Subramanian et al., 2003; Subramanian et al. , 2001; Varma et al, 2008), real options (Davis and Owens, 2003; Rogers et al, 2002; Wang and Hwang, 2007), and stochastic programming. For brevity, we limit our discussion to SP for new product development, which are most relevant to the work presented in this paper.

Schmidt and Grossmann (1996) formulated a SP, and solved the deterministic equivalent MILP model for the scheduling of testing tasks in the agricultural and pharmaceutical industries. Maravelias and Grossmann (2001) develop a Lagrangean-relaxation based heuristic to solve the integrated new product development and batch scheduling problem. Varma et al. (2007) introduced a Lagrangean-relaxation based heuristic to solve the pharmaceutical R&D pipeline version of the new product/process development problem originally presented in Blau et al. (2004). The authors showed greater than 30% improvement on strategies using a priority based system. Colvin and Maravelias (2008) developed a multistage stochastic program to solve the clinical trial scheduling portion of the pharmaceutical R&D pipeline management problem. The authors solved larger instances of the problem using a non-anticipativity constraint reduction approach (Colvin and Maravelias, 2009), a rolling horizon approach (Colvin and Maravelias, 2009), and a branch-and-cut algorithm (Colvin & Maravelias, 2010). The branch-and-cut algorithm gradually enforced constraints as they are violated. Solak et al. (2010) used a sample average approximation approach, where candidate solutions of the R&D pipeline management problem were generated using subsets of the full problem space.

3.2.1.3 Literature Combining the Production Planning, Capacity Planning and R&D Pipeline Management Problems

Most available literature is limited to either integrated production and capacity planning problems or integrated production and new technology development problems. Tsang et al. (2007) considered an application of new product development and capacity planning for a multi-product case study for vaccine production in the pharmaceutical industry. The authors formulated the problem using a two-stage stochastic program where investment decisions (i.e. production selection and manufacturing site selection and construction) are made in the first stage. In the second stage, recourse actions such as production decisions are taken. Yang et al. (2014) used a flexible capacity approach to address demand uncertainty. The flexible capacity approach allows the firm to realize demand before production decisions are made to avoid waste. The authors modeled technology investments using a quadratic production cost. To solve the model, they used a three-step decision making process to maximize profits. Fahmi and Cremaschi (2013) developed a simulation-optimization approach to study the evolution of different technology pathways in the biomass to commodity chemical system with uncertainty in technology performance and product demand. Fahmi et al. (2014) developed a deterministic model which extended the model originally presented in Fahmi and Cremaschi (2013). The authors performed sensitivity analysis to determine the impact of the model parameters on the solution. In this work, we develop a multistage stochastic programming formulation which determines investment decisions (i.e. capacity expansion and research and development) considering uncertainty in the technology development process, as well as, uncertainty in the demand.

3.2.2 The NTIP Problem Model Components

3.2.2.1 The Problem Statement

The NTIP problem is defined as follows. Given,

(1) a fixed length planning horizon divided into equal time periods [$t \in \boldsymbol{T}$],

(2) a set of chemical processing technologies [$i \in \boldsymbol{I}$],

(3) a set of chemicals [$n \in \boldsymbol{N}$], and

(4) a set of technology maturity stages [$sg \in \boldsymbol{SG}$],

the objective of the problem is to determine the technology investment schedule which provides the expected minimum production cost. We assume that investment decisions can either be to develop a new technology or to expand the capacity of an existing technology. Investment in new technologies are assumed to be limited with a fixed level monetary resource. To ensure that demand at each time period is met, we assume that a product can either be produced using existing capacity or can be purchased at market price.

The total cost of production for a desired product can be represented as the sum of the raw material costs (RMC_i) and the costs of technology development and expansion (DEC_i) (Eq. 3.2)

$$C = \sum_i RMC_i + DEC_i \tag{3.1}$$

The raw material cost includes the cost of feedstocks and materials used to produce intermediates. Costs for technology development and expansion are incurred either by direct research investment in technology improvement or by increasing its capacity. For this work, we model the cost as the feedstock cost, however the model can be easily modified to include operational costs as a function of production rate. Research investment costs are calculated as the sum of all investments in any technology used in the production of a product. Investments in capacity expansion are estimated by multiplying the cost of unit capacity expansion by the total capacity expansion. The cost of unit capacity expansion is assumed to be represented by a two-factor learning curve.

3.2.2.2 Two Factor Learning Curve to Model Unit Capacity Expansion Cost

The concept of learning curves was first introduced in Wright (1936). The author observed that the number of hours required to produce one unit decreased at a constant rate as the number of items produced doubled. In general, the function of the learning-curve model is to link the capacity, the labor hours, or the output to the unit cost. The learning curve developed by Wright (1936) is considered a single factor learning curve. Single factor learning curves relate a single observation (e.g., the labor hours) to the unit cost (e.g., unit manufacturing time). The factor associated with single factor learning curves is called the learning-by-doing factor.

In two-factor learning curves, in addition to the learning-by-doing factor, a learning-by-searching factor, which generally models the impact of R&D on reduction of cost, is incorporated. Kouvaritakis et al. (2004) used a two-factor learning curve approach to model the cost of unit capacity expansion for large scale energy models. Fahmi et al. (2014) used a two-factor learning curve to model the cost of unit capacity expansion for renewable energy technologies. Here, we consider a two-factor learning curve of the form shown in Eq. 3.2.

$$CapC_i = CapC_{i,0}\left(\frac{Cap_i}{Cap_{i,0}}\right)^B\left(\frac{CRD_i}{CRD_{i,0}}\right)^A \tag{3.2}$$

The two-factor learning curve shown in Eq. 3.2 uses the initial cost of capacity expansion ($CapC_{i,0}$) and two-factors. The first factor, $\left(\frac{Cap_i}{Cap_{i,0}}\right)^B$, is the ratio of the current capacity of the technology (Cap_i) to the initial capacity ($Cap_{i,0}$). The factor represents the reduction in the cost of unit capacity expansion due to learning-by-doing. The coefficient B is called the elasticity of the learning-by-doing factor. The elasticity is a measure of the impact the factor of interest has on the cost of unit capacity expansion. The second factor, $\left(\frac{CRD_i}{CRD_{i,0}}\right)^A$, is the ratio of the cumulative research investment level (CRD_i) and the initial research investment level ($CRD_{i,0}$). The second factor represents the cost reduction associated with learning-by-searching. The factor is modified with an elasticity A.Figure 3.3 plots the learning-by-doing and the learning-by-searching ratios versus the factor value for a sample case with a learning-by-doing elasticity of -0.21 and a learning-by-searching elasticity of -0.07. In Figure

3.3, the curves for each factor start at one and gradually decrease as the ratio increases. When the elasticity is smaller (i.e. -0.07) the rate at which the factor value decreases is lower.

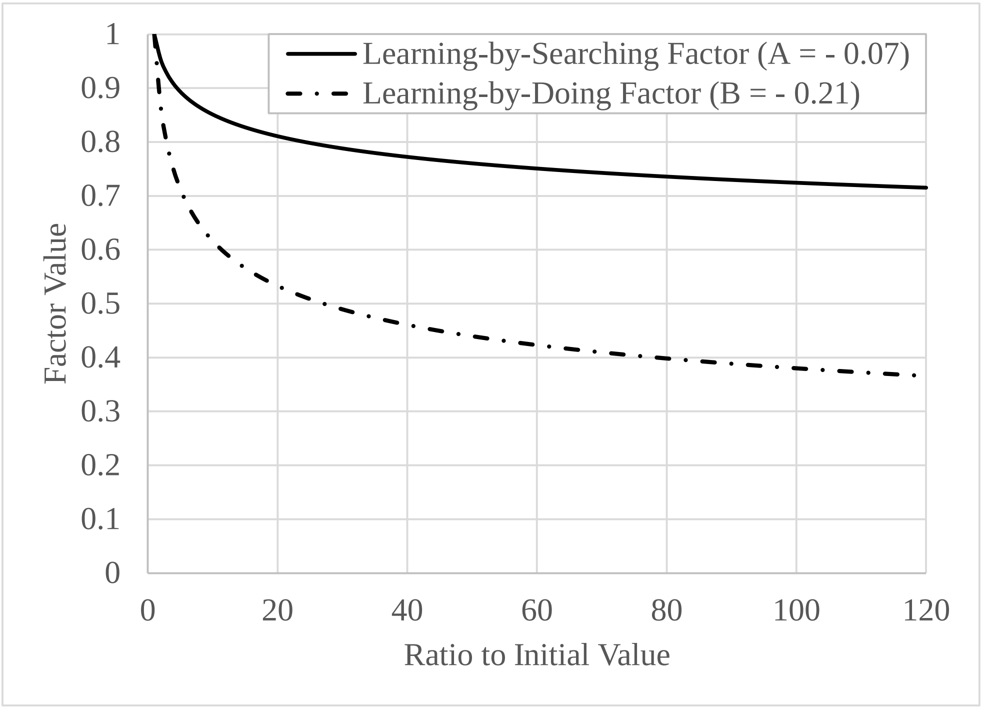


Figure 3.3 Plot of the Learning-By-Doing and the Learning-By-Searching Ratios vs. Factor Value

In the new product development process, the elasticity values for new technologies are not known with certainty. The values that the elasticities take are known to be strictly negative between 0 and -1, and are realized after capacity expansion and research investments are made. To represent the uncertainty in the values of the elasticities for each new potential technology, we introduce uncertain parameters α and β to represent the values of A and B in the formulation of the two factor learning curve. In the model, the values of the initial capacity cost, the initial R&D expenditures, and the initial installed capacity is assumed to be known.

3.2.2.3 Network Representation of the CPI

Flow of material through the processes or chemical process industry is generally modeled using network representations. The most commonly used network representations are state-task networks (STN), state-equipment networks, or resource task networks (Floudas and Lin, 2005). State-task networks provide an intuitive approach to modeling material flow (Floudas and Lin, 2005). For NTIP model, the material flow is represented using a STN. In a STN, there are two types of nodes: state nodes which contain information about the state of the chemical, such as composition, pressure, temperature, and/or flowrate, and task nodes which contain process information (Floudas and Lin, 2005). The STN uses a predefined set of directed arcs to connect the chemical states to the tasks (processes).

For the NTIP model, the state nodes in the network represent feedstocks, intermediates, and products. Task nodes represent different processes. Arcs connect state nodes and task nodes to relate material flow between chemical states and processes. Flow within the network is constrained using a set of mass balances written around the state nodes. In the model, we do not allow storage of intermediates and feedstocks. This implies that all intermediates produced must also be consumed in the time period in which they are produced. We also prohibit the sale of intermediates. The only way to obtain feedstocks or intermediates is either purchasing or production. To ensure that demand is always met, we assume that the product may be purchased from another manufacturer if not produced at enough quantities. Figure 3.4 shows an example of a state task network with three technologies (tasks) and four chemicals. In Fig. 3.4, CHEM 1 and CHEM 2 represent feedstock chemicals. The only way to procure CHEM 1 and CHEM 2 is to purchase them (i.e., there are no arrows entering CHEM 1 or CHEM 2). CHEM 3 is considered an intermediate because it can either be purchased or produced by TECH 1 or TECH 2, and it is not the desired product. CHEM 4, for this example, is the desired product, and it may be purchased or produced using TECH 3.

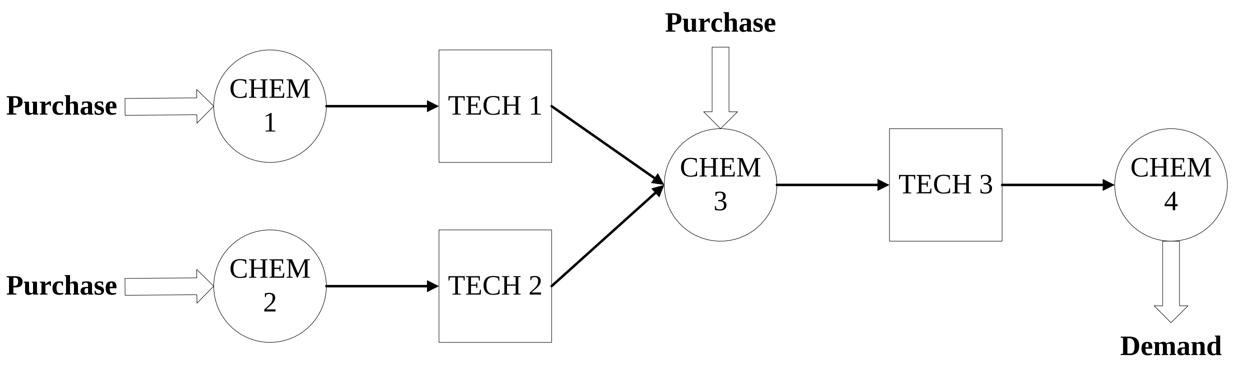


Figure 3.4 An example STN for the NTIP problem.

Mathematically, we can represent the material flow through the network in Fig. 3.4 using a set of balances written around each chemical (CHEM 1-4). The general form of the balances is shown in Eq. 3.3, where $M_{store,n}$, $M_{purchase,n}$, $M_{produce,n}$ and $M_{consume,n}$ represent the level of storage of chemical n, the amount of chemical n purchased, the amount of chemical n produced, and the amount of chemical n consumed. The demand of chemical n is defined by δ_n.

$$M_{store,n} = M_{purchase,n} + M_{produce,n} - M_{consume,n} - \delta_n \quad \forall n \in \boldsymbol{N} \tag{3.3}$$

For feedstock nodes, CHEM 1 and CHEM 2 in Fig. 3.4, we can reduce Eq. 3.3 to Eq. 3.4 by setting storage, production and demand variables equal to zero.

$$0 = M_{purchase,n} - M_{consume,n} \quad \forall n \in \boldsymbol{N}_{Feedstocks} \tag{3.4}$$

Intermediate chemicals can be consumed, produced and purchased. Therefore, we can reduce equation 3 for CHEM 3 in Fig. 3.4 to Eq. 3.5.

$$0 = M_{purchase,n} + M_{produce,n} - M_{consume,n} \quad \forall n \in \boldsymbol{N}_{Intermediates} \tag{3.5}$$

The final equation is the balance is for the product, CHEM 4, which can be purchased, produced, or used to meet the demand. Reducing Eq. 3.3 results in Eq. 3.6.

$$0 = M_{purchase,n} + M_{produce,n} - \delta_n \quad \forall n \in \boldsymbol{N}_{Products} \tag{3.6}$$

In addition to the balances written in Eqs. 3.4- 3.6, the consumption and production of some chemicals are related through the technologies connecting them. For example, the production of CHEM 3 consumes CHEM 1, and this relationship is defined in Eq. 3.7.

$$M_{produce,CHEM3} = \frac{\nu_{CHEM3} \cdot MW_{CHEM3}}{\nu_{CHEM1} \cdot MW_{CHEM1}} \cdot \eta_{TECH1} \cdot M_{consume,CHEM1} \tag{3.7}$$

Equation 3.7 uses the stoichiometric coefficient (ν) from the reaction occurring in technology (TECH 1) and the yield of the technology (η) to calculate the amount of CHEM 3 produced. In cases where there may be no reaction occurring in the technology, the yield (η) can be seen as an efficiency of the technology and the stoichiometric ratio becomes a mass ratio of the product to the feed.

3.2.2.4 Representation of the Technology Development Process

The development of new technology is a time consuming and expensive process. The cost of introducing a new technology varies by sector. Generally, introducing a new technology into the CPI requires hundreds of millions of dollars after accounting for R&D and process development costs (Miremadi et al., 2013). New technology development can be considered a portion of the new product/process development process (PDP). The PDP does not require that the product be new to the market; it may be an existing commodity chemical.

The PDP has been studied extensively. A review of relevant product development research is presented in Krishnan and Ulrich (2001). Representation of the PDP depends on the types of decisions in the planning. In this work, we emphasize the decision between competing projects where either the product or the production route is different. The five-stage generalized PDP presented in Cooper (1990), shown in Fig. 3.5, is adopted for this work. The first stage, the Preliminary Assessment Stage, provides an initial technical and market assessment for the new product. Technical assessments during this stage gauge the product's manufacturing feasibility and the estimated time and cost of production. The Definition Stage is the second stage. The technical portion of this phase provides a more detailed technical analysis of the project. The analysis examines the do-ability of the project and ensures that the project is economically viable.

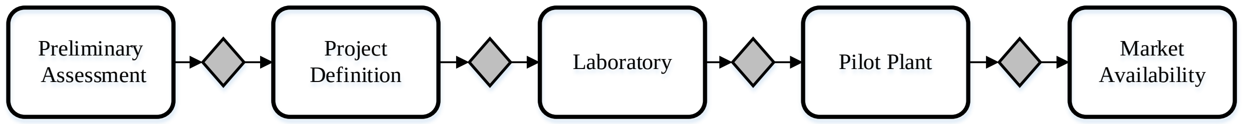


Figure 3.5 The five-stage PDP presented in Cooper (1990)

The Development Stage or Laboratory, is the first heavy spending stage. In this stage, the product is developed. Development of the product in the CPI often requires extensive laboratory experimentation. Detailed plans for operation are also developed during this stage. After the Laboratory Stage, the product proceeds to the Validation or Pilot Plant stage. The validation stage tests the feasibility of producing the product in a relevant environment. The test includes verification of the processes used through pilot production, field testing, and/or in-house testing. The final stage is the Market Availability, or commercialization, stage. This stage implements the market launch plan, and the production and operation plans.

The PDP can be represented using a stage-gate framework. The stage-gate framework creates a discrete set of stages. Each stage corresponds to one of the technology development stages. The stage-gate framework has been successfully applied to product development in several different industries including the development of medical devices (Pietzsch et al, 2009), project management in the tech industry (Conforto and Amaral, 2016), new feedstock development in the chemical process industry (Fahmi and Cremaschi, 2012b), sustainability planning (Tingstro, 2006), and the pharmaceutical R&D pipeline planning (Subramanian et al., 2003,Colvin and Maravelias, 2008). Seider et al. (2009) gives a perspective on the use of the stage-gate process for product design.

The first two stages of the PDP consist of idea generation and viability studies. The overall cost of these stages is negligible compared to the cost of the last three stages, which are related to process development and commercialization. In this work, we assume that all new technology development projects considered have completed the first two stages of the PDP and that projects still need to complete the laboratory and pilot plant stages before reaching the commercialization stage. The current stage of a technology is determined based on its installed capacity. Furthermore, we assume that each stage can be associated with a pre-defined minimum capacity. Figure 3.6 shows the stage-gate framework used for this work.

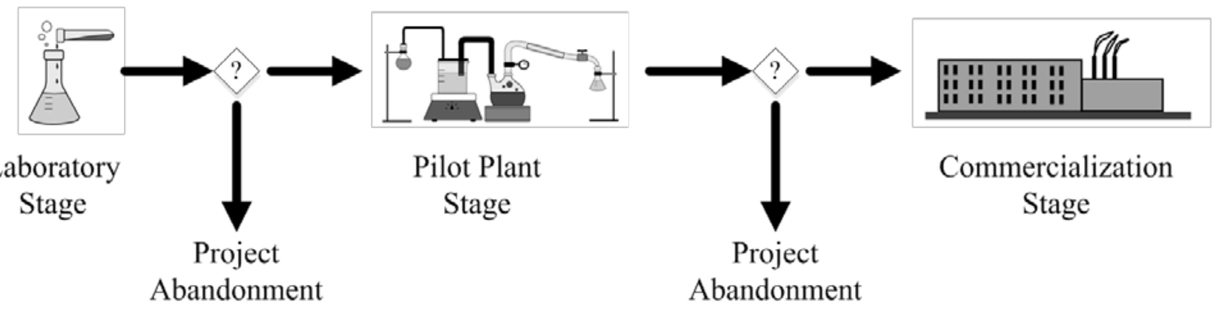


Figure 3.6 The stage-gate representation of the general NTIP problem

The actual yield of a technology is not known until the technology reaches the commercialization stage. To reach a stage, the installed capacity of a technology must reach the minimum capacity threshold for that stage. After a technology completes a stage, an assessment is performed to determine whether or not investment in the technology should continue. Whether or not a project will be abandoned is not known with certainty until the previous stage is completed. In this work, we represent the uncertainty in project abandonment at each stage using a random Bernoulli trial with known probability, ρ_{sg}, where the possible outcomes are either project abandonment (PA) or continued investment (CP). Assuming that the decision of project abandonment is independent of the previous stage realizations, we can combine the uncertainty in each stage of the PDP and use a single uncertain parameter, ψ. Figure 3.7(A) shows the four possible outcomes of a series of two Bernoulli trials, representing the laboratory and pilot plant stages. Once a project is abandoned, investment in later stages ceases. This implies that outcomes can be aggregated. Colvin and Maravelias (2008) used a similar approach to aggregate uncertain parameter outcomes in the clinical trial planning problem. Figure 3.7(B) shows the aggregated form of the outcomes, $\psi \in \{I/PA,$ II/PA, II/CP}.

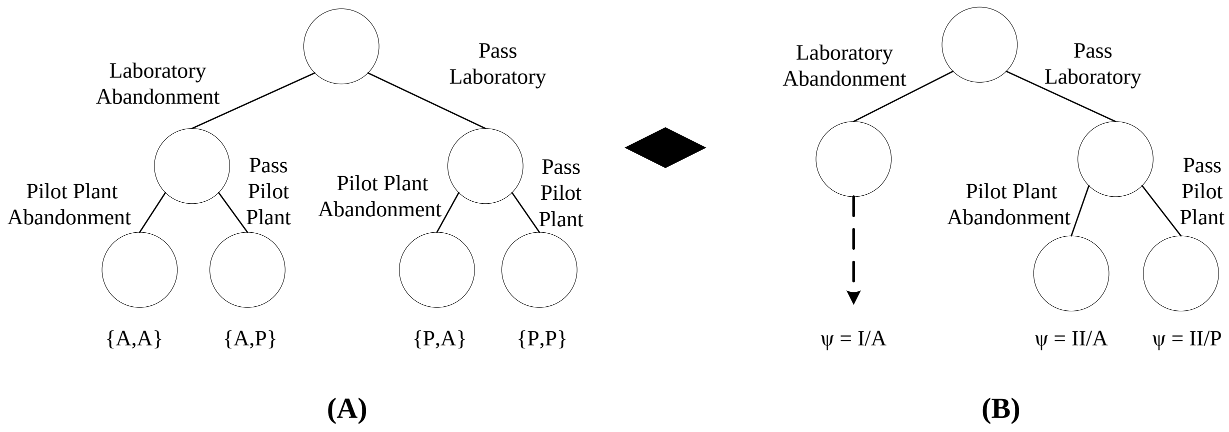


Figure 3.7 Uncertainty representation in the NTIP product development process

Once the technology reaches the commercialization stage, the underlying value for the yield of the technology is realized. Technologies that are in the commercialization stage at the beginning of the planning horizon are assumed to be fully mature, and hence, their capacity expansion costs and yields are known with certainty and do not change throughout the planning horizon.

3.2.2.5 Scenario Representation

Scenarios are obtained using the Cartesian product of the realizations of the uncertain parameters. The number of uncertain parameters in this problem depends on the number of technologies below commercialization stage at the beginning of the planning horizon (i.e., new technologies), and the length of the planning horizon. There are four uncertain parameters associated with each new technology. The first two uncertain parameters, ψ and χ, are associated with the outcome of technology development (ψ) and the yield of the process (χ). The other two uncertain parameters are related to the values of A and B represented by α and β in the two-factor learning curve (Eq. 2.1). These parameters define how the capacity expansion cost may change with investments. The demand for the desired product is not known with certainty, and for each time period beyond first, there is an uncertain parameter representing the demand of the desired product(s).

Because the uncertain parameter ψ is discrete, we can represent the uncertainty exactly. Unlike ψ, the parameters α, β, yield (χ), and the demand at each time period (D_t) are continuous uncertain parameters. For each continuous uncertain parameter, it is assumed that the parameter can be approximated by a set of discrete values. These values represent the set of potential realizations for the uncertain parameter. As an example, consider the installed capacity elasticity, β, of a new technology. The value of the elasticity is not known with certainty but there is a set of possible outcomes that are known. The set $\beta \in \{\beta_1, \beta_2, \dots, \beta_k\}$ represents all k possible outcomes. This definition yields $|\mathbf{U}|$ sets, where $|\mathbf{U}|$ is the number of uncertain parameters. The n-fold Cartesian product of these sets yields the scenario set. For a single new technology where the planning horizon is three years divided into three equal time periods, the uncertain parameters would be ψ representing the success of developing the technology, and, α and β representing the elasticities of the two-factor learning curve. Additionally, the product demand throughout the planning horizon, denoted by D_2 and D_3 for time periods two and three, is uncertain. The n-fold Cartesian product $\psi \times \alpha \times \beta \times D_2 \times D_3 \times \chi$ yields the scenario set, i.e., the set of n-tuples with $|\psi| \cdot |\alpha| \cdot |\beta| \cdot |D_2| \cdot |D_3| \cdot |\chi|$ number of elements. Each n-tuple in this set represents a unique combination of the outcomes of the uncertain parameters. There are three possible outcomes for ψ. Assuming that there are two possible outcomes for each of the remaining uncertain parameters, a high and a low value, the total number of scenarios would be equal to $3 \cdot 2^4$or 48 for a problem with a single new technology and a planning horizon of three time periods.

3.2.3 The Multi-stage stochastic programming formulation of the NTIP problem

3.2.3.1 The Objective Function

The objective of the NTIP problem is to minimize the expected total cost (*ETC*) of production over the planning horizon. This cost is calculated via Eq. 3.8.

$$ETC = \sum_{s} p_s(MatCst_s + CapExCst_s + RDCst_s) \tag{3.8}$$

In Eq. 3.8, the expected total cost of production, ETC, is equal to the probability of each scenario occurring, p_s, multiplied by the total production cost of each scenario s. Total production cost for each scenario includes the total raw material costs ($MatCst_s$), total capacity expansion costs ($CapExCst_s$), and the total R&D costs ($RDCst_s$).

The total raw material cost includes expenditures to procure all materials, i.e., feedstock, intermediates, and final products (if necessary) to meet the demands, and is calculated using Eq. 3.9 for each scenario.

$$MatCst_s = \sum_n \sum_t cd_t \cdot MCst_{n,t} \cdot F_{n,t,s} \qquad \forall s \tag{3.9}$$

At each time period, the unit price of chemical n ($/tonne) is given as a constant value, $MCst_{n,t}$. Material cost for chemical n at time period t is calculated as the product of the amount required, $F_{n,t,s}$, and its unit price. The material cost at each time period is then discounted using the factor cd_t to account for the time value of money. The factor cd_t is defined as $(1 + IR)^{1-t}$ where IR is the discount rate. The total raw material cost of each scenario is the sum of all the material costs over the planning horizon.

The R&D costs are direct investments into improving technologies. The total research investment of each scenario is simply the sum of all research investments ($RD_{i,t,s} - RD_{i,t-1,s}$) into all technologies. Equation 3.10 shows how the total R&D cost for each scenario is calculated.

$$RDCst_s = \sum_i \sum_t cd_t \cdot (RD_{i,t,s} - RD_{i,t-1,s}) \qquad \forall s \tag{3.10}$$

The final term in the objective function is the total capacity expansion cost. This cost represents the investments for installing or increasing the production capacity of technologies. Here, we assume that the capacity expansion cost of technology i at time t under scenario s can be estimated using its unit capacity-expansion cost ($CC_{i,t,s}$) ($\$/kg$) and level of capacity expansion $\left(X_{i,t,s}\right)$. Then, the total capacity expansion cost for each scenario is calculated by multiplying the capacity expansion cost of technology i at time t by the discounting factor for the time value of money, and by summing the costs over all technologies and time periods (Eq. 3.11).

$$CapExCst_s = \sum_i \sum_t 1000\, cd_t \cdot CC_{i,t,s} \cdot X_{i,t,s} \quad \forall s \tag{3.11}$$

3.2.3.2 The Unit Capacity-Expansion Cost

Section 3.2.2. discussed the use of a two-factor learning curve to describe how the cost of unit capacity expansion changes with the cumulative capacity and R&D expenditures. The current cumulative installed capacity is linked to the capacity expansion at each time period using Eq. 3.12. The level of capacity expansion is defined as the difference between the cumulative installed capacity of technology i at the current time period (t) under scenario s, $CX_{i,t,s}$, and the cumulative installed capacity at the previous time period, $CX_{i,t-1,s}$.

$$X_{i,t,s} = CX_{i,t,s} - CX_{i,t-1,s} \qquad \forall i,t,s \tag{3.12}$$

Equation 3.13 shows the cost of unit capacity expansion for technology i at time t under scenario s as a function of the initial capacity expansion cost, $CC0_i$, the initial and current installed capacities, and the initial and current total R&D investments.

$$CC_{i,t,s} = CC0_i \left(\frac{CX_{i,t,s}}{CX_{i,0}}\right)^{\beta_{i,t,s}} \left(\frac{RD_{i,t,s}}{RD_{i,0}}\right)^{\alpha_{i,t,s}} \qquad \forall i,t,s \tag{3.13}$$

The rates of cost reductions due to capacity expansion (learning-by-doing elasticity) and research investments (learning-by-searching elasticity), given as $\beta_{i,t,s}$ and $\alpha_{i,t,s}$, are not known with certainty at the time of the investments. To incorporate this uncertainty, Eqs. 3.14 and 3.15 are used to represent $\beta_{i,t,s}$ and $\alpha_{i,t,s}$.

$$\beta_{i,t,s} = \beta_{i,s} NN^{\beta}_{i,t,s} \tag{3.14}$$

$$\alpha_{i,t,s} = \alpha_{i,s} NN^{\alpha}_{i,t,s} \tag{3.15}$$

In Eq. 3.14, $\beta_{i,t,s}$ either takes a value of 0 when the value of β is unknown or $\beta_{i,s}$ after the true value of β_i is realized. The binary variable $NN^{\beta}_{i,t,s}$ takes a value of one if the realization of β_i has occurred and zero otherwise. The value for $\alpha_{i,t,s}$ is calculated using the same approach as $\beta_{i,t,s}$. Whether the realization of α_i has occurred is represented using a binary variable $NN^{\alpha}_{i,t,s}$. The value of $\alpha_{i,s}$ represent the scenario specific realization of the value of α_i. Inserting Eqs. (3.14) and (3.15) into Eq. (3.13), and applying log transformation to the resulting expression yields Eq. 3.16.

$$\log CC_{i,t,s} = \beta_{i,s} NN^{\beta}_{i,t,s} \log\left(\frac{CX_{i,t,s}}{CX_{i,0}}\right) + \alpha_{i,s} NN^{\alpha}_{i,t,s} \log\left(\frac{RD_{i,t,s}}{RD_{i,0}}\right) + \log(CC0_i) \tag{3.16}$$

3.2.3.3 Constraints Related to Material Flow in CPI

The topology of the CPI is modeled using a state-task network. Each technology/process in the CPI is represented with a task network, and the state nodes represent the feedstocks, intermediates, and products. To track material flow between each technology, material balances are written around state nodes. The general mass balance for each state node in the NTIP problem is given in Eq. 3.17

$$0 = F_{n,t,s} - D_{n,t,s} + G_{n,t,s} \qquad \forall n, t, s \tag{3.17}$$

Eq. 3.17 states that, for chemical n at time t for scenario s, the amount purchased, $F_{n,t,s}$, combined with the amount produced, $G_{n,t,s}$, must be equal to the demand, $D_{n,t,s}$. The amount produced is calculated using Eq. 3.18.

$$G_{n,t,s} = \sum_i g_{i,n,t,s} = \sum_i \gamma_{i,n,PR} \chi_{i,t,s} M_{i,PR,t,s} \quad \forall\, n, t, s \tag{3.18}$$

Notice that the total production of chemical n, $G_{n,t,s}$ (kmol), is calculated by summing over the amount produced by each technology i, $g_{i,n,t,s}$. We represent the production of each technology using a mass ratio $(\gamma_{i,n,PR})$, the yield $(\chi_{i,t,s})$, and the amount of primary reactant/feedstock used in technology i at time t $(M_{i,PR,t,s})$. For technology i, the mass ratio is given as the ratio of the stoichiometric coefficient of the chemical n and the primary reactant/feedstock *PR*, multiplied by the ratio of the molecular weights $(MW)\left(\gamma_{i,n,PR} = \frac{v_{i,n} \cdot MW_n}{v_{i,PR} \cdot MW_{PR}}\right)$ (Section 3.1.2.3). Equation 3.19 calculates the conversion for technology i.

$$\chi_{i,t,s} = Y_{i,COM,t,s} \cdot \chi_{i,s} \quad \forall i, t, s \tag{3.19}$$

The conversion of a technology i before the commercialization stage (COM) is considered to be zero (it is assumed that product cannot be used for satisfying demand before reaching the commercialization stage). In Eq. 3.19, the value $\chi_{i,s}$ is the realized conversion for technology i at the commercialization stage. The binary variable $Y_{i,COM,t,s}$ becomes one if technology i has reached the commercialization stage at time t or before under scenario s.

3.2.3.4 Formulation of the Technology Development Process

Whether or not technology i has successfully completed stage sg at time t under scenario s is represented by a binary variable $Y_{i,sg,t,s}$. The stage of a technology is determined based on the cumulative installed capacity. Equations 3.20 and 3.21 establish the value of $Y_{i,sg,t,s}$ using the current cumulative installed capacity, $CX_{i,t,s}$. The parameter ${CX_{i,sg}}^{min}$ represents the minimum installed capacity of a technology in stage sg. Equation 3.21 uses the variable $Y_{i,sg,t,s}$ to ensure that the installed capacity does not exceed the minimum capacity of the next development stage. Equation 3.22 enforces that for a technology to be in stage sg, it must have completed stage $(sg - 1)$. Equation 3.23 relates the technology stages along the planning horizon. Eq. 3.23 states that if a technology has completed a stage at $(t - 1)$ then the technology must have also completed the stage at time t. This constraint ensures that the technology stage only increases along the planning horizon.

$$CX_{i,t,s} \geq {CX_{i,sg}}^{min} \cdot Y_{i,sg,t,s} \qquad \forall i, sg, t, s \tag{3.20}$$

$$CX_{i,t,s} \leq \sum_{sg<3} \left({CX_{i,sg+1}}^{min} - {CX_{i,sg}}^{min} \right) Y_{i,sg,t,s} \qquad \forall i,t,s \tag{3.21}$$

$$Y_{i,sg,t,s} \leq Y_{i,sg-1,t,s} \qquad \forall i, sg > 1, t, s \tag{3.22}$$

$$Y_{i,sg,t,s} \leq Y_{i,sg,t-1,s} \qquad \forall i, sg, t > 1, s \tag{3.23}$$

To ensure that production is less than the current installed capacity in the commercialization stage, the constraint shown in Eq. 3.24 is introduced. Equation 3.24 bounds the maximum amount of production and ensures that the production of a technology is zero before the technology reaches the final stage. In Eq. 3.24, $\theta_{i,s}$ is a binary parameter representing the successful completion of the technology development process. Recall, ψ represents the outcome of the technology development. In scenarios where the uncertain parameter ψ_i has a realization of II/CP, the parameter $\theta_{i,s}$ takes a value of one. Otherwise, the value of the parameter is zero.

$$M_{i,n,t,s} \leq Y_{i,3,t,s} CX_{i,t,s}\, \theta_{i,s} \quad \forall i,t,s \tag{3.24}$$

3.2.3.5 Non-Anticipativity Constraints

The MSSP formulation of the NTIP problem introduces separate decision variables for each scenario, and hence, non-anticipativitiy constraints (NACs) should be added explicitly to the formulation to ensure that decisions are identical before the realization of uncertain parameters. The first step in developing the NACs is to identify when two scenarios differ. The decisions for both scenarios must be identical until the time the differentiating event occurs. If the uncertain parameter that differentiates two scenarios is exogenous, the time of differentiation is known *a priori*. For instance, the demand of a product at time period t will be revealed at the end of time-period t. Therefore, the decision-variable values for the capacity expansion of each technology ($X_{i,t,s}$) must be identical for scenarios s and s' that only differ in their demand realizations until the differentiating time period ($t_{s,s'}^{diff}$). Mathematically, this is represented with the equality constraint shown in Eq. 3.25.

$$X_{i,t,s} = X_{i,t,s'} \quad \forall i, t < t_{s,s'}^{diff} \tag{3.25}$$

For the case of endogenous uncertain parameters, however, $t^{diff}$is not known *a priori* and depends on when the decisions associated with the differentiating events are made. For example, the value for the uncertain parameter β_i for technology i is only realized after there is an expansion in the capacity of technology i. This means that all decisions for scenarios s and s', where scenario s and s' differ only in the realization of β_i, must be identical until the capacity of technology i is expanded sufficient number of times. We can represent this using a disjunction as shown in Eq. 3.26.

$$\begin{bmatrix} X_{i,t,s} = X_{i,t,s'} \\ NN_{i,t,s}^{\beta} \end{bmatrix} \vee \left[\neg NN_{i,t,s}^{\beta} \right] \quad \forall i, t, s \tag{3.26}$$

The binary variable $NN_{i,t,s}^{\beta}$ takes a value of one when there has been sufficient number of investments for expansion of capacity in technology i at time t under scenario s and a value of zero otherwise. We can write the binary disjunction using big-M constraints. Equation 3.27 shows the form of these constraints for the case of the elasticity parameter β. The big-M constraints are written for the set $\boldsymbol{\phi_\beta}$ which contains all (s, s') which differ only in the realization of β_i.

$$-M \cdot NN_{i^{s,s'},t,s}^{\beta} \leq X_{i,t,s} - X_{i,t,s'} \leq \; M \cdot NN_{i^{s,s'},t,s}^{\beta} \quad \forall i, t < t_{s,s'}^{diff}, (s, s') \in \boldsymbol{\phi_\beta} \tag{3.27}$$

An appropriate value for M can be calculated as the maximum difference between the decision variables. In this case, the value for M would be the maximum level of capacity expansion (Cap_i^{Max}). To minimize the number of NACs, the NAC reduction approaches introduced in Goel and Grossmann (2004), Goel et al. (2006), and Goel and Grossmann (2006) are used.

The SP formulation of the NTIP problem contains three decision variables for each scenario, the cumulative installed capacity ($CX_{i,t,s}$), the cumulative research expenditures ($RD_{i,t,s}$), and the amount of each chemical generated/consumed ($M_{i,n,t,s}$) by each technology. The problem also has a recourse action variable which describes the amount of material purchased ($F_{i,n,t,s}$). It is important to note that NACs only need to be written for decision variables. The formulation also contains four different uncertain parameters, $\psi_i, \alpha_i, \beta_i$, and D_t. For each uncertain parameter k, we generate sets, $\boldsymbol{\phi_k}$, which include scenario pairs which are differentiable in outcome by k. For the exogenous uncertain parameter demand (D_t), the differentiating time period is known *a priori*. This implies the NACs can be written using the form shown in Eq.3.25. These NACs are given in Eqs. 3.28-3.29.

$$CX_{i,t,s} = CX_{i,t,s'} \qquad \forall i, t < t^{diff}_{s,s'}, (s, s') \in \boldsymbol{\phi_D} \tag{3.28}$$

$$RD_{i,t,s} = RD_{i,t,s'} \qquad \forall i, t < t^{diff}_{s,s'}, (s, s') \in \boldsymbol{\phi_D} \tag{3.29}$$

Equations 3.28-3.29 ensures that the decision variables ($CX_{i,t,s}$ and $RD_{i,t,s}$) are equal to each other for scenarios s and s' before the time period of differentiation $\left(t^{diff}_{s,s'}\right)$.

For endogenous uncertain parameters, the differentiating time period depends on when differentiating event(s) occurs. In the NTIP problem, there are four endogenous uncertain parameters, ψ_i, χ_i α_i, and β_i. For each endogenous uncertain parameter, we introduce a binary variable similar to the one for β_i in Eq. 3.27. The binary variable represents whether or not the differentiating event has occurred.

The differentiating event for ψ_i (whether or not the project is abandoned) occurs when the technology reaches the minimum capacity of the next stage, e.g., the technology may or may not be abandoned after the laboratory investigations are completed. The value of binary variable, $Y_{i,sg,t,s}$, can be used to represent the differentiating event for ψ_i. To generate the NACs associated with ψ_i, the same procedure outlined for the example with β_i is used. This requires the generation of a disjunction and subsequent conversion of the disjunction into a set of Big-M constraints. For each parameter, a NAC is written for each decision variable, and they can be seen in Eqs. 3.30-3.32.

$$-CX_{i,3}^{max} Y_{i^{s,s'},sg^{s,s'},t,s} \leq CX_{i,t,s} - CX_{i,t,s'} \leq Y_{i^{s,s'},sg^{s,s'},t,s} CX_{i,3}^{max} \quad \forall i, t, (s, s') \in \boldsymbol{\phi_\psi} \tag{3.30}$$

$$-|\boldsymbol{T}| RD^{max} Y_{i^{s,s'},sg^{s,s'},t,s} \leq RD_{i,t,s} - RD_{i,t,s'} \leq Y_{i^{s,s'},sg^{s,s'},t,s} RD^{max} |\boldsymbol{T}| \quad \forall i, t, (s, s') \in \boldsymbol{\phi_\psi} \tag{3.31}$$

$$-CX_{i,3}^{max} Y_{i^{s,s'},sg^{s,s'},t,s} \leq M_{i,n,t,s} - M_{i,n,t,s'} \leq Y_{i^{s,s'},sg^{s,s'},t,s} CX_{i,3}^{max} \quad \forall i, t, (s, s') \in \boldsymbol{\phi_\psi} \tag{3.32}$$

In Eqs. 3.30-3.32, the big-M values are set using information from the NTIP problem. For each decision variable, we calculate big-M values by finding the theoretical maximum differences between variable values in different scenarios. To illustrate this, consider Eq. 3.30 where the installed capacity is constrained. In one scenario, the installed capacity may have not been expanded leaving the cumulative installed capacity at its initial value, $(CX0_i)$; however, in another scenario, the capacity of a technology may have been expanded in order to produce enough product to meet all the demand. The difference in the installed capacity between the two scenarios represents the maximum difference in the installed capacity. Thus, this value is used as the big-M value. A similar approach is used for Eq. 3.31. Using the same logic of Eq. 3.30, we can calculate the maximum total research investment, RD^{max}, as the sum over all time periods of the maximum research investment. Equation 3.35 constrains the maximum difference in the production amounts of chemical n by technology i at time t between two scenarios. The big-M value is the maximum possible cumulative installed capacity, $(CX_{i,3}^{max})$. We calculate this value using the logic used in Eqs. 3.30 and 3.31.

The disjunctions of non-anticipativity are constructed for endogenous parameters χ_i, α_i and β_i. The realization of the process yield (χ_i) occurs after a technology reaches the commercial stage of development. In order to determine when this event occurs we use the decision variable $Y_{i,COM,t,s}$, which indicates whether technology i at time t is in the commercial stage for scenario s or not. Non-anticipativity is ensured using big-M constraints shown in Eqs. 3.33 -3.35.

$$-CX_{i,3}^{max} Y_{i^{s,s'},3,t,s} \leq CX_{i,t,s} - CX_{i,t,s'} \leq Y_{i^{s,s'},3,t,s} CX_{i,3}^{max} \quad \forall i,t,(s,s') \in \boldsymbol{\phi_\chi} \tag{3.33}$$

$$-|\boldsymbol{T}| RD^{max} Y_{i^{s,s'},3,t,s} \leq RD_{i,t,s} - RD_{i,t,s'} \leq Y_{i^{s,s'},3,t,s} RD^{max} |\boldsymbol{T}| \quad \forall i,t,(s,s') \in \boldsymbol{\phi_\chi} \tag{3.34}$$

$$-CX_{i,3}^{max} Y_{i^{s,s'},3,t,s} \leq M_{i,n,t,s} - M_{i,n,t,s'} \leq Y_{i^{s,s'},3,t,s} CX_{i,3}^{max} \quad \forall i,t,(s,s') \in \boldsymbol{\phi_\chi} \tag{3.35}$$

In the case of α_i, the value of the parameter is realized after investments are made in research expenditures, and the value of β_i is realized after investments in capacity expansion are made. Here we introduce two binary variables, $N_{i,t,s}^{\alpha}$and $N_{i,t,s}^{\beta}$, which take a value of one if an investment has been made for research expenditures and for capacity expansion in technology i at time period t in scenario s, respectively. The value of $N_{i,t,s}^{\alpha}$and $N_{i,t,s}^{\beta}$ are calculated using Eqs. 3.36 and 3.37.

$$RD_{i,t,s} - RD_{i,t-1,s} \geq {N_{i,t,s}}^{\alpha} \quad \forall i,t,s \tag{3.36}$$

$$CX_{i,t,s} - CX_{i,t-1,s} \geq {N_{i,t,s}}^{\beta} \quad \forall i,t,s \tag{3.37}$$

The value of RD^{max} and X_i^{max} represent the maximum research investment and capacity expansion investment at each time period. In order to relate the variables, $N_{i,t,s}^{\alpha}$and $N_{i,t,s}^{\beta}$, to the realization of the uncertain parameters α and β, we use $NN_{i,t,s}^{\alpha}$and $NN_{i,t,s}^{\beta}$. The function of these binaries is to indicate when sufficient numbers of investments have been made for differentiation scenarios. We define $NN_{i,t,s}^{\alpha}$and $NN_{i,t,s}^{\beta}$ using Eqs. 3.38 and 3.39.

$$\sum_{t'<t} N_{i,t',s}{}^{\alpha} \leq NN_{i,t,s}{}^{\alpha} - 1 + V^{\alpha} \quad \forall i,t,s \tag{3.38}$$

$$\sum_{t'<t} N_{i,t',s}{}^{\beta} \leq NN_{i,t,s}{}^{\beta} - 1 + V^{\beta} \quad \forall i,t,s \tag{3.39}$$

Equations 3.38 and 3.39 relate the investment decisions at each time period ($N_{i,t,s}^{\alpha}$and $N_{i,t,s}^{\beta}$) to the total number of investments needed to realize the uncertain parameters α_i and β_i (V^{α} and V^{β}). After representing the differentiating event with binary variables, disjunctions are used to write the NACs associated with α and β. The non-anticipativity constraints for α_i can be seen in Eqs. 3.40-3.42.

$$-CX_{i,3}^{max} NN_{i^{s,s'},t,s}^{\alpha} \leq CX_{i,t,s} - CX_{i,t,s'} \leq NN_{i^{s,s'},t,s}^{\alpha} CX_{i,3}^{max} \quad \forall i,t,(s,s') \in \boldsymbol{\phi_\alpha} \tag{3.40}$$

$$-|\boldsymbol{T}| RD^{max} NN_{i^{s,s'},t,s}^{\alpha} \leq RD_{i,t,s} - RD_{i,t,s'} \leq NN_{i^{s,s'},t,s}^{\alpha} |\boldsymbol{T}| RD^{max} \quad \forall i,t,(s,s') \in \boldsymbol{\phi_\alpha} \tag{3.41}$$

$$-CX_{i,3}^{max} NN_{i^{s,s'},t,s}^{\alpha} \leq M_{i,n,t,s} - M_{i,n,t,s'} \leq NN_{i^{s,s'},t,s}^{\alpha} CX_{i,3}^{max} \quad \forall i,t,(s,s') \in \boldsymbol{\phi_\alpha} \tag{3.42}$$

Big-M values for Eqs. 3.40-3.42 are the same as the big-M values used in Eqns. 34-3.35. Each of the NACs in Eqs. 3.40-3.42 are written for the set of scenarios (s, s') which differ only in the realized value of α_i ($\boldsymbol{\phi_\alpha}$).

Equations 3.43-3.45 are the Big-M representations of the disjunctions written for the uncertain parameter β_i. Big-M values are identical to the ones used in Eqns. 34-3.35. We write the constraints for the set of scenarios (s, s') which differ only in the realized value of β_i ($\boldsymbol{\phi_\beta}$).

$$-CX_{i,3}^{max} NN_{i^{s,s'},t,s}^{\beta} \leq CX_{i,t,s} - CX_{i,t,s'} \leq NN_{i^{s,s'},t,s}^{\beta} CX_{i,3}^{max} \quad \forall i, t, (s, s') \in \boldsymbol{\phi_{\beta}} \tag{3.43}$$

$$-|\boldsymbol{T}| RD^{max} NN_{i^{s,s'},t,s}^{\beta} \leq RD_{i,t,s} - RD_{i,t,s'} \leq NN_{i^{s,s'},t,s}^{\beta} |\boldsymbol{T}| RD^{max} \quad \forall i, t, (s, s') \in \boldsymbol{\phi_{\beta}} \tag{3.44}$$

$$-RD^{max} NN_{i^{s,s'},t,s}^{\beta} \leq M_{i,\mathrm{n},t,s} - M_{i,\mathrm{n},t,s'} \leq NN_{i^{s,s'},t,s}^{\beta} RD^{max} \quad \forall i, t, (s, s') \in \boldsymbol{\phi_{\beta}} \tag{3.45}$$

The objective function of the model, the minimum expected total cost of production, is given in Eq. 3.8. The decision variables in the formulation include the installed capacity ($CX_{i,t,s}$), the level of R&D investment ($RD_{i,t,s}$), the production level ($M_{i,n,t,s}$), and the amount of material purchased ($F_{i,n,t,s}$). Decision variables are constrained using four different sets of equations. The four sets of equations represent production and capacity planning constraints (Eqs. 3.17-3.19), process development constraints (Eqs. 3.21-3.24), economic constraints (Eqs. 3.9-3.15), and NACs (Eqs. 3.28-3.45). The resulting optimization model is a large scale non-convex MINLP.

3.2.3.6 An Approximate Linear Model for Tight Upper Bounds

The nonlinear terms in the NTIP problem are in Eqs. 3.11, 3.13, and 3.18. Equation 3.11 contains bilinear terms, i.e., the product of the capacity expansion cost and the capacity expansion. Equation 3.13 contains two exponential terms and the product of the two exponential terms. The non-linear term in Eq. 3.18 is the product of $Y_{i,Commercial,t,s}$ and the molar amount produced ($M_{i,n,t,s}$). We applied a combination of approaches to remove and approximate these nonlinear terms with linear ones. Specifically, the bilinear terms in Eq. 3.11 and the bilinear term formed by the product of the exponential terms in Eq. 3.13 are approximated using linearly-segmented tight relaxations (Misener et al., 2011). The exponential terms are approximated using tight piece-wise linear lower and upper estimators (Fahmi and

Cremaschi, 2015) and the non-linear term in Eq. 3.18 is removed using exact linearization (Muhittin and Ossama, 1992).

The detailed steps of the linearization of the bilinear terms in Eqs. 3.11 using linearly-segmented tight convex under-estimators and concave over-estimators (Misener et al., 2011) are shown below. The resulting constraints are given in Eqs. 3.46-3.55, and these constraints replace Eqs. 3.11 in the original model. We first define a new continuous variable, $CCX_{i,t,s} = CC_{i,t,s}X_{i,t,s}$, and substitute this variable into Eq. 3.11. This substitution yields Eq. 3.46.

$$CapExCst_s = \sum_i \sum_t CCX_{i,t,s} \quad \forall s \tag{3.46}$$

We select the variable $X_{i,t,s}$ to partition, and assume that three is a sufficient number of partitions to accurately represent $X_{i,t,s}$. Equation 3.47 shows the calculation of the length of each segment, a_X.

$$a_X = \frac{X_i^{Max}}{3} \tag{3.47}$$

The range of variable $X_{i,t,s}$ is between zero (corresponding to no capacity expansion at time t for technology i under scenario s) and X_i^{Max} , which is the maximum allowable capacity expansion at each time period. Equation 3.48 introduces a new binary variable, $\lambda_{i,t,s,np}^X$, which indicates whether or not the value of $X_{i,t,s}$ is in the partition np. The summation shown in Eq. 3.48 ensures that $X_{i,t,s}$ is only in one partition.

$$\sum_{np \in \{1,2,3\}} \lambda_{i,t,s,np}^X = 1 \quad \forall i,t,s \tag{3.48}$$

The partition that $X_{i,t,s}$ exists in is calculated using the inequalities given in Eq.3.49. Since the partitions are equally spaced, the value of $X_{i,t,s}$ should fall between the lower value of the partition and the upper value of the partition.

$$\sum_{np\in\{1,2,3\}} a_X \cdot (np-1) \cdot \lambda^X_{i,t,s,np} \le X_{i,t,s} \le \sum_{np\in\{1,2,3\}} a_X \cdot np \cdot \lambda^X_{i,t,s,np} \quad \forall i,t,s \tag{3.49}$$

The value of $CC_{i,t,s}$ is calculated using Eq. 3.50. Equation 3.50 segments the value of $CC_{i,t,s}$ into a minimum value, $CC^{Min}_{i,t,s}$, plus a segment value, $\Delta CC_{i,t,s,np}$. The value for $CC^{Min}_{i,t,s}$ represents the minimum value that $CC_{i,t,s}$ can take, and is calculated using Eq. 3.13 with the scenario specific realizations of $\alpha_{i,s}$ and $\beta_{i,s}$.

$$CC_{i,t,s} = CC^{Min}_{i,t,s} + \sum_{np} \Delta CC_{i,t,s,np} \quad \forall i,t,s \tag{3.50}$$

By definition $\Delta CC_{i,t,s,np}$ must be strictly non-negative. The maximum for $\Delta CC_{i,t,s,np}$ is calculated based on the maximum difference in capacity expansion cost in each partition. The inequalities generated from these bounds is shown in Eq. 3.51.

$$0 \le \Delta CC_{i,t,s,np} \le \left(CC0_i - CC^{Min}_{i,t,s}\right) \cdot \lambda^X_{i,t,s,np} \quad \forall i,t,s,np \tag{3.51}$$

The tight upper and lower estimators for $CCX_{i,t,s}$ are defined using Eqs. 3.52-3.55.

$$CCX_{i,t,s} \ge X_{i,ts} \cdot CC^{Min}_{i,t,s} + \sum_{np\in\{1,2,3\}} \left(a_X \cdot (np-1)\right) \cdot \Delta CC_{i,t,s,np} \quad \forall i,t,s \tag{3.52}$$

$$\begin{aligned} CCX_{i,t,s} &\ge X_{i,t,s} \cdot CC0_i \\ &\quad + \sum_{np\in\{1,2,3\}} (a_X \cdot np) \cdot \left(\Delta CC_{i,t,s,np} - \left(CC0_i - CC^{Min}_{i,t,s}\right)\lambda^X_{i,t,s,np}\right) \quad \forall i,t,s \end{aligned} \tag{3.53}$$

$$CCX_{i,t,s} \le X_{i,t,s} \cdot CC^{Min}_{i,t,s} + \sum_{np\in\{1,2,3\}} (a_X \cdot np) \cdot \Delta CC_{i,t,s,np} \quad \forall i,t,s \tag{3.54}$$

$$\begin{aligned} CCX_{i,t,s} &\le X_{i,t,s} \cdot CC0_i \\ &\quad + \sum_{np\in\{1,2,3\}} \left(a_X \cdot (np-1)\right)\left(\Delta CC_{i,t,s,np} - \left(CC0_i - CC^{Min}_{i,t,s}\right) \cdot \lambda^X_{i,t,s,np}\right) \quad \forall i,t,s \end{aligned} \tag{3.55}$$

To remove the nonlinear terms of Eq. 3.13, we start by defining two new variables as $NF^{\beta}_{i,t,s} = \left(\frac{CX_{i,t,s}}{CX_{i,0}}\right)^{\beta_{i,t,s}}$ and $NF^{\alpha}_{i,t,s} = \left(\frac{RD_{i,t,s}}{RD_{i,0}}\right)^{\alpha_{i,t,s}}$. Substituting Eqs. 3.14 and

3.15 for $\alpha_{i,t,s}$ and $\beta_{i,t,s}$ yields Eqs. 3.56 and 3.57 for $NF^{\beta}_{i,t,s}$ and $NF^{\alpha}_{i,t,s}$. Here, we will detail the procedure for generating tight linear upper and lower estimators for the factor $NF^{\beta}_{i,t,s}$; the process for $NF^{\alpha}_{i,t,s}$ is identical. (The equation set that define tight linear upper and lower estimators for $NF^{\alpha}_{i,t,s}$ are in Appendix B.)

$$NF^{\beta}_{i,t,s} = \left(\frac{CX_{i,t,s}}{CX_{i,0}}\right)^{\beta_{i,s} {NN_{i,t,s}}^{\beta}} \quad \forall i,t,s \tag{3.56}$$

$$NF^{\alpha}_{i,t,s} = \left(\frac{RD_{i,t,s}}{RD_{i,0}}\right)^{\alpha_{i,s} {NN_{i,t,s}}^{\alpha}} \quad \forall i,t,s \tag{3.57}$$

Because ${NN_{i,t,s}}^{\beta}$ is a binary variable, Eq. 3.58 is equivalent to Eq. 3.56.

$$NF^{\beta}_{i,t,s} = \left(1 - {NN_{i,t,s}}^{\beta}\right) + {NN_{i,t,s}}^{\beta} \left(\frac{CX_{i,t,s}}{CX_{i,0}}\right)^{\beta_{i,s}} \tag{3.58}$$

Expanding Eq. 3.58 yields Eq. 3.59. Notice that Eq. 3.59 contains an exponential term. We use linearly segmented upper and lower estimators to linearize this term. (Fahmi and Cremaschi, 2015)

$$NF^{\beta}_{i,t,s} = 1 - {NN_{i,t,s}}^{\beta} + {NN_{i,t,s}}^{\beta} \left(\frac{1}{CX_{i,0}}\right)^{\beta_{i,s}} \left(CX_{i,t,s}\right)^{\beta_{i,s}} \tag{3.59}$$

To obtain linearly segmented upper and lower estimators, we first define the exponential term using a continuous variable, $CX\beta 1_{i,t,s} = \left(CX_{i,t,s}\right)^{\beta^{1}_{i,s}}$ and substitute the new variable into Eq. 3.59 to obtain Eq. 3.60.

$$NF^{\beta}_{i,t,s} = 1 - {NN_{i,t,s}}^{\beta} + {NN_{i,t,s}}^{\beta} \left(\frac{1}{CX_{i,0}}\right)^{\beta_{i,s}} CX\beta 1_{i,t,s} \tag{3.60}$$

We divide the base variable, $CX_{i,t,s}$, into m partitions of equal lengths. Note that different partition lengths and number of partitions for base variable and each exponent can also be defined to tighten the estimators, if necessary. Here, we use m

= 3. The domain of the variable, $CX_{i,t,s}$, is represented as a series of m segments of length a_i^β.

$$a_i^\beta = \frac{CX_i^{Max} - CX_i^{Min}}{3} \quad \forall i \tag{3.61}$$

To determine which partition each variable is in, binary variable, $b_{i,t,s,np}^\beta$, is introduced. This binary variable takes a value of one if the variable is currently in the associated partition and zero, otherwise. Equation 3.62 ensures that only one partitions is active for each variable, and Eq. 3.63 bounds the installed capacity ($CX_{i,t,s}$) within the appropriate partition values.

$$\sum_{np\in\{1,2,3\}} b_{i,t,s,np}^\beta = 1 \quad \forall i,t,s \tag{3.62}$$

$$\begin{aligned} CX_i^{Min} + \sum_{np\in\{1,2,3\}} a_i^\beta \cdot (np-1) \cdot b_{i,t,s,np}^\beta &\le CX_{i,t,s} \\ &\le CX_i^{Min} + \sum_{np\in\{1,2,3\}} a_i^\beta \cdot np \cdot b_{i,t,s,np}^\beta \quad \forall i,t,s \end{aligned} \tag{3.63}$$

For each partition, the tangent line at its beginning yields the lower bound. Because the exponents, i.e., elasticities of the two-factor learning curve, are negative, the slope of the exponential term is strictly decreasing as the base variable increases. This property ensures that the tangent line at the beginning of the partition always produces a value lower than the value of the exponential within the partition. Equation 3.64 shows the lower estimators for $\left(CX_{i,t,s}\right)^{\beta_{i,s}}$.

$$\begin{aligned} CX\beta 1_{i,t,s} \ge \beta_{i,s}^1 \left(CX_{i,t,s}^{Min} + a_i^\beta (np-1)\right)^{\beta_{i,s}-1} \left(CX_{i,t,s} - \left(CX_{i,t,s}^{Min} + a_i^\beta (np-1)\right)\right) \\ + \left(CX_{i,t,s}^{Min} + a_i^\beta (np-1)\right)^{\beta_{i,s}} \qquad \forall i,t,s,np \in \{1,2,3\} \end{aligned} \tag{3.64}$$

We obtain the upper estimator by connecting the function values at the beginning and end of the partition with a line. Again, the strictly decreasing nature of the exponential function with negative exponent ensures that the upper estimators always provide a value that is higher than the exponential function. The upper estimator for the $\left(CX_{i,t,s}\right)^{\beta_{i,s}}$is shown in Eq. 3.65.

$$CX\beta1_{i,t,s} \leq \sum_{np\in\{1,2,3\}} \frac{\left(CX_{i,t,s}^{Min} + a_i^\beta \cdot np\right)^{\beta_{i,s}} - \left(CX_{i,t,s}^{Min} + a_i^\beta (np-1)\right)^{\beta_{i,s}}}{a_i^\beta} bCX_{i,t,s,np} - b_{i,t,s,np}^\beta \left(CX_{i,t,s}^{Min} + a_i^\beta \cdot np\right) \frac{\left(CX_{i,t,s}^{Min} + a_i^\beta \cdot np\right)^{\beta_{i,s}} - \left(CX_{i,t,s}^{Min} + a_i^\beta (np-1)\right)^{\beta_{i,s}}}{a_i^\beta} + b_{i,t,s,np}^\beta \left(CX_{i,t,s}^{Min} + a_i^\beta \cdot np\right)^{\beta_{i,s}} \quad \forall i,t,s \tag{3.65}$$

Notice that the upper estimator depends on the partition. We use the binary variable, $b_{i,t,s,np}^\beta$, to determine the active upper estimator line, which results in a non-linear term due to the multiplication of the binary variable with the continuous variable $CX_{i,t,s}$. We use exact linearization for removing this nonlinearity (Muhittin and Ossama, 1992). Replacing the product of $b_{i,t,s,np}^\beta$ and $CX_{i,t,s}$ with a continuous variable, $bCX_{i,t,s}$, in Eq. 3.66 and bounding this new variable via Eqs. 3.66 - 3.68 removes this nonlinearity.

$$bCX_{i,t,s,np} \leq b_{i,t,s,np} CX_{i,t,s}^{Max} \quad \forall i,t,s,np \tag{3.66}$$

$$bCX_{i,t,s,np} \geq b_{i,t,s,np} CX_{i,t,s}^{Min} \quad \forall i,t,s,np \tag{3.67}$$

$$bCX_{i,t,s,np} \leq CX_{i,t,s} \quad \forall i,t,s,np \tag{3.68}$$

Finally, the nonlinear term resulting from the multiplication of the binary variable, $NN_{i,t,s}^\beta$, and the continuous variable, $CX\beta1$, at the exponent in Eq. 3.60 is linearized. We define this product with a continuous variable, $NCX\beta1_{i,t,s}$, substitute this variable into Eq. 3.60, and bound $NCX\beta1_{i,t,s}$ with Eqs. 3.70-3.72.

$$NF^{\beta}_{i,t,s} = 1 - NN^{\beta}_{i,t,s} + \left(\frac{1}{CX_{i,0}}\right)^{\beta_{i,s}} NCX\beta 1_{i,t,s} \quad \forall i,t,s \tag{3.69}$$

$$NCX\beta 1_{i,t,s} \le NN^{\beta}_{i,t,s} \cdot CX\beta 1^{Max}_{i,t,s} \quad \forall i,t,s \tag{3.70}$$

$$NCX\beta 1_{i,t,s} \ge NN^{\beta}_{i,t,s} \cdot CX\beta 1^{Min}_{i,t,s} \quad \forall i,t,s \tag{3.71}$$

$$NCX\beta 1_{i,t,s} \le CX\beta 1_{i,t,s} \quad \forall i,t,s \tag{3.72}$$

After linearizing the exponential terms, we substitute the continuous variables, $NF^{\beta}_{i,t,s}$ and $NF^{\alpha}_{i,t,s}$, into Eq. 3.13. This substitution yields Eq. 3.73.

$$CC_{i,t,s} = CC_{i,0} NF^{\beta}_{i,t,s} NF^{\alpha}_{i,t,s} \quad \forall i,t,s \tag{3.73}$$

A linear approximation of Eq. 3.73 is obtained using linearly-segmented tight relaxations (Misener et al., 2011), and the resulting constraint set is given in Appendix B.

Substituting Eq. 3.19 into Eq. 3.18 yields Eq. 3.74, in which the binary variable $Y_{i,COM,t,s}$ is multiplied by the continuous variable $M_{i,PR(i),t,s}$. To linearize product of a continuous variable and a binary variable, we use exact linearization (Muhittin and Ossama, 1992). We start by replacing the product with a continuous variable $ZM_{i,PR,t,s}$ and adding appropriate bounds for the new continuous variable, $ZM_{i,PR,t,s}$. The bounding constraints are given in Eqs. 3.75-3.77

$$G_{n,t,s} = \sum_{i} \gamma_{i,n,PR} \chi_{i,s} Z_{i,COM,t,s} M_{i,PR(i),t,s} \quad \forall\, n,t,s \tag{3.74}$$

$$ZM_{i,PR,t,s} \le Z_{i,COM,t,s} \cdot PD^{Max}_{i} \tag{3.75}$$

$$ZM_{i,PR,t,s} \ge 0 \tag{3.76}$$

$$ZM_{i,PR,t,s} \leq M_{i,PR(i),t,s} \tag{3.77}$$

Replacing Eq. 3.11 with Eqs. 3.46-3.55, Eq. 3.13 with Eqs. 3.61-3.68 and Eqs. 3.70-3.73, and Eq. 3.18 with Eqs. 3.74-3.77 of the original NTIP problem formulation yields a large scale MILP, which is a tight relaxation of the original MINLP.

3.2.4 Case Studies

To demonstrate how the proposed MSSP model can be used to support new technology investment planning decisions, we present and discuss the results of three case studies. Each case study considers a different number of developing technologies and a different network architecture. The MINLP and the MILP relaxation models of the case studies were implemented using PYOMO 4.1 (Hart et al., 2012), and solved using Auburn University Hopper Cluster. We used BARON 17.1 (Tawarmalani and Sahinidis, 2005) as the MINLP solver, IPOPT 3.12 (Belotti, Lee, Liberti, Margot, & Wächter, 2009) as the nonlinear programming (NLP) solver, and CPLEX 12.63 as the MILP and LP solver. The solutions of the MILP relaxations provide a tight lower bound for the MINLP problems. For case studies where MINLP problems were not solved to optimality within 5%, the values of the integer variables were fixed to the relaxed MILP solution, and the resulting NLP problem was solved to obtain a feasible solution and an upper bound.

3.2.4.1 Case Study 1- Considering a New Technology

The first case study compares a new non-mature technology, TECH 1, and a mature technology, TECH 2. Both technologies can be used to produce the same product, CHEM 3 (Fig. 3.8).

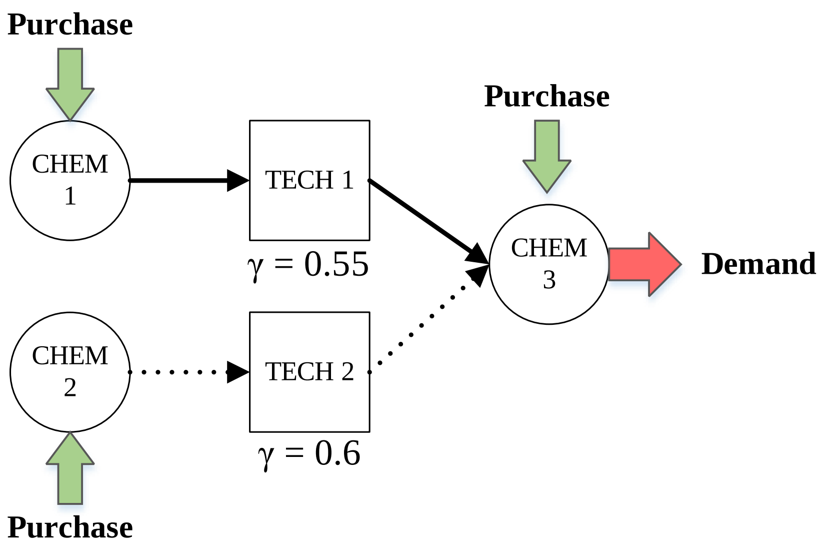


Figure 3.8 The network topology for Case Study 1

In the network, CHEM1 can be converted to CHEM3 using developed TECH1, or CHEM2 can be converted to CHEM3 using under-development TECH2. For this case study, the investment decisions should be made along a planning horizon of three year-long time periods. The conversion of chemicals that occur in each technology and their corresponding γ values are shown in below the technology boxes in Fig. 3.15. Despite the simple network topology and the short planning horizon, Case Study 1 has seven uncertain parameters. We assume that all uncertain parameters, besides those associated with project abandonment, have two realizations, a High and a Low value. The possible realizations for each of these parameters (Elasticity Parameters(α and β), Yield (χ), and Demand (D)) can be found in Tables 3.8 and 3.9.

Table 3.8 The possible realizations of α, β, and χ for Case Study 1.

Technology	α Values		β Values		χ Values	
	High	Low	High	Low	High	Low
TECH1	N/A		N/A		0.85	
TECH2	-0.18	-0.20	-0.06	-0.08	0.98	0.95

Table 3.9 The possible realizations of demand in Case Study 1

Chemical	Demand Values (Mtonnes)			
	2		3	
	High	Low	High	Low
CHEM1	0		0	
CHEM2	0		0	
CHEM3	23.5	28	22.1	26

Given the possible realizations of the parameters shown in Tables 3.8 and 3.9 and the outcomes of project abandonment for each technology, the number of scenarios in this problem is calculated to be $3 \cdot 2^5 = 96$. Values of the remaining parameters are summarized in Table 3.10.

Table 3.10 Parameters for Case Study 1

Parameter	Value		
	TECH1	***TECH2***	
Maximum Capacity Expansion (Mtonnes)	6	6	
Initial R&D Investment (Billion Dollars)	1	5	
Initial Installed Capacity (Mtonnes)	1	2.5	
Initial Capacity Expansion Cost (\$/kg)	1	1.4	
	CHEM1	***CHEM2***	***CHEM3***
Raw Material Cost (\$/tonne)	724	845	1200
Molecular Weight (kg/kmol)	50	62	72
	TECH2, Laboratory	***TECH2, Pilot Plant***	
Probability of Success	95%	99%	

The MINLP model of Case Study 1 has 1728 binary variables, 9409 continuous variables, and 46,561 constraints. In 10,000 CPU s, BARON Version 17.1 completed two iterations and was not able to find a feasible solution. The MILP model, which is a relaxation of the original model, has 6336 binary variables, 13,153 continuous variables and 64,705 constraints, and was solved to optimality by CPLEX version 12.63 in 4.08 CPU s. The optimum objective value of the MILP was found to be \$147 billion. The MILP solution was used to fix the integer variables in the MINLP, yielding a NLP. The NLP had 7680 variables and 46,560 constraints, was solved to optimality by IPOPT Version 3.12 in 12.4 CPU s. The optimum was \$188 billion, which yielded a gap of 21%.

Figure 3.9 plots the empirical cumulative distribution functions (CDFs) of the minimum total costs of MILP and NLP optimum solutions.

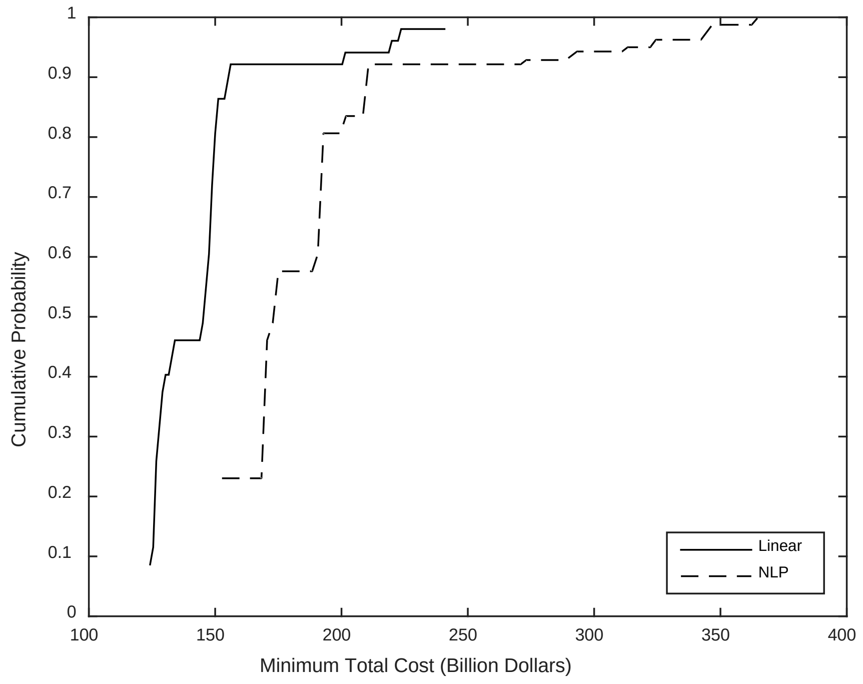


Figure 3.9 The empirical cumulative distribution functions of minimum total cost for MILP and NLP optimum solutions for Case Study 1. The MILP model is a tight relaxation of the original MINLP formulation, the NLP model is obtained by fixing the integer variables of the original MINLP to the solution of MILP model.

The true CDF would lie between the MILP and the NLP CDFs shown in Fig. 3.9. To illustrate the expansion and production strategies, we discuss the capacity expansion decisions of two scenarios for MILP and NLP solutions. In the first scenario, TECH1 successfully completes the laboratory and pilot-plant stages, and reaches the commercial stage. For the second scenario, the project is abandoned (i.e., fails) at the pilot plant stage. At the first time period, both scenarios have identical decisions. The MILP solution recommends expanding TECH2 capacity by 2 Mtonnes and investing 1 million dollars in research. The level of capacity expansion for developed TECH1 is 6 Mtonnes. After the first time period, the two scenarios become differentiable. In the second time period, the scenario where TECH2 is successfully developed, capacity expansions to both TECH1 and TECH2 are observed. The solution favors TECH1 by expanding it by the maximum allowable level (6 Mtonnes). A capacity expansion of 5.885 Mtonnes is recommended for TECH2 in time period two. At time period three, TECH2 capacity is increased by 1.8 Mtonnes in order to meet the entirety of the observed demand. The capacity of TECH1 does not change in the third time period. In contrast, for the second scenario where TECH2 fails to complete pilot plant stage successfully, the capacity of TECH1 is expanded at a level of 6 Mtonnes during the second and third time periods. At each time period in both scenarios, the demand is met using the currently installed capacity and any demand that cannot be met is made up by purchasing the product.

The NLP solution is identical to the MILP solution. By fixing the integer variables in the MINLP, the solution is fixing the decisions on whether or not to expand or invest. This implies that the decision variables will be very similar to the MILP case. The only difference is in the value of the objective function. The NLP does not approximate the capacity expansion cost, as such the NLP solution has an objective function value higher than that of the MILP solution.

3.2.4.2 Scaling of the NTIP Problem

We consider two additional hypothetical case studies to study how the NTIP MSSP model scales with the number of under-development technologies and network size. Case Study 2 compares two under-development technologies (TECH1 and TECH2) for producing the same chemical from different feedstocks (Fig. 3.10(A)). Case Study 3 considers a retrofitting problem, which consists of a larger network with two under-development technologies (Fig. 3.10(B)). In Case Study 2, the product (CHEM3) can either be produced using CHEM1 and TECH1 or CHEM2 and TECH2. The planning horizon is three year-long time periods. For each under-development technology, there are four uncertain parameters. For Case Study 2, we assume that there are two possible outcomes for α, β, and χ, similar to Case Study 1. The values for the realizations of the uncertain parameters and the fixed parameters of Case Study 2 are compiled in Tables B.1-B.5 in Appendix B. The number of scenarios for Case Study 2 is $3^2 \cdot 2^8 = 2304$, which is a 24-factor increase when compared to Case Study 1.

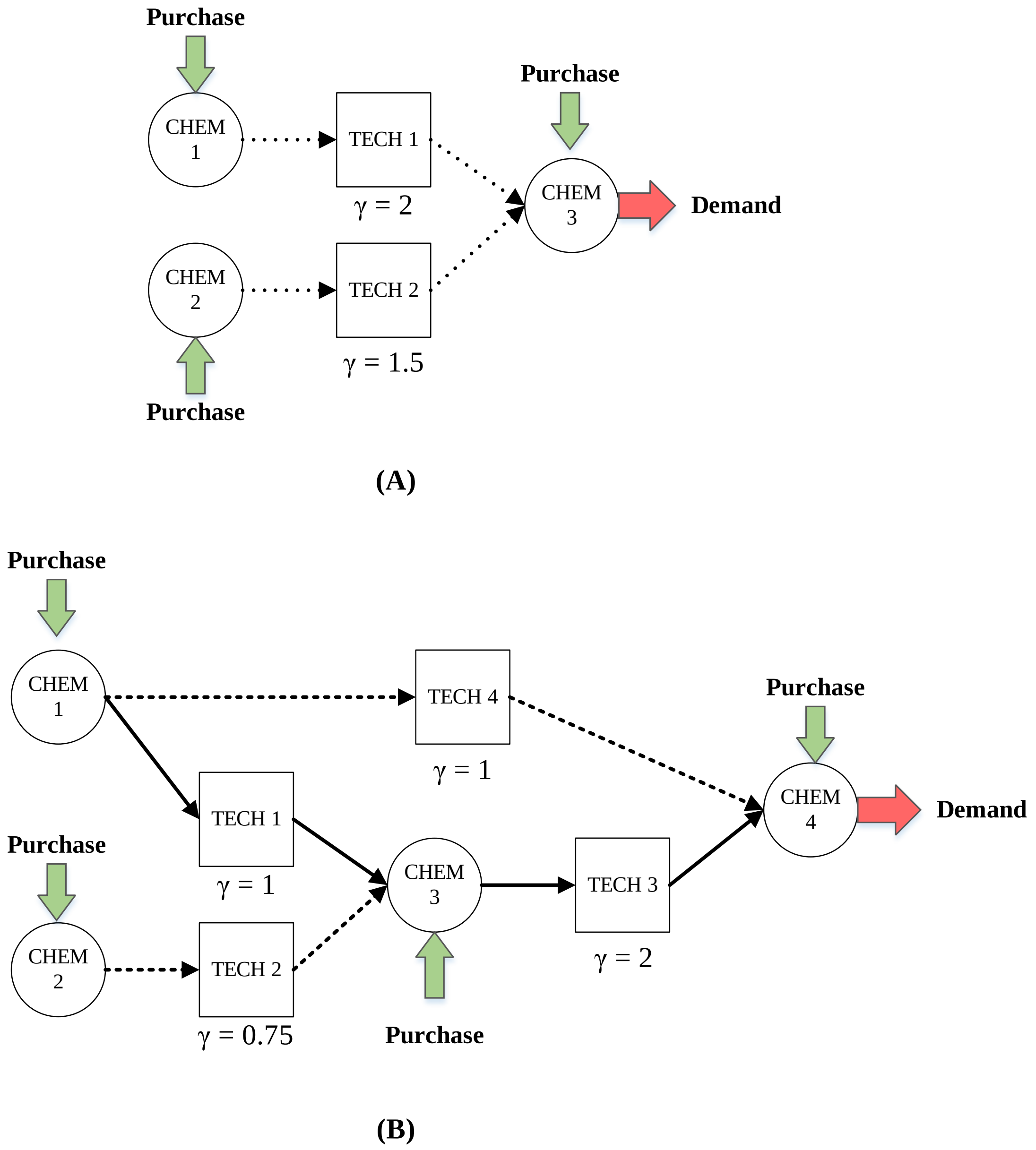


Figure 3.10 The state-task networks for Case Studies 2 (A) and 3 (B)

In Case Study 3 (Fig. 3.10(B)), the network has a total of four technologies (TECH1-TECH4) and five chemicals (CHEM1-CHEM4). There are two under-development technologies, TECH2 and TECH4, and each of the uncertain parameters, α, β, and χ has two realizations. Further, we assume that the demand at each time period will also take one of two values. The planning horizon is four years divided into one-year increments. The MSSP of Case Study 3 has nine uncertain parameters and $3^2 \cdot 2^9 = 4{,}608$ scenarios. The values of the Case Study 3 fixed parameters and uncertain parameters can be found in Tables B.6-B.10 in Appendix B.

The deterministic equivalents of the MSSP for Case Study 2 has 343,297 variables, 41,472 of which are binary variables, and has 1.884 million constraints. Case Study 3 has 294,912 binary variables, 3.866 million continuous variables, and 7.963 million constraints. For both MINLP problems, Baron Version 17.1 did not complete an iteration in 10,000 CPU s.

The MILP model of Case Study 2 has 262,336 binary variables, 481,537 continuous variables, and 2.755 million constraints. CPLEX Version 12.6.3 solved the MILP to an optimality gap of 1% within 1074 CPU s, and yielded an optimum solution of \$91.3 billion dollars. After solving the MILP, we generated a NLP by fixing the values of the binary variables to the MILP solution. The resulting NLP problem had 301,825 continuous variables and 1.884 million constraints. IPOPT Version 3.12 failed due to lack of available allocated memory for the linear equation solver, MA27 (the linear equation solver compiled with IPOPT). We gradually increased the factor that changes the size of the workspace allocated for solving the linear equations from its default value of 5 to a value of 10000, but failed to obtain a solution to the NLP problem.

The solution of the MILP for Case Study 2 recommended to expand capacities of TECH1 1 Mtonne and TECH2 1 Mtonne at first time period. The solution also recommends investing 1 million in researching each technology. This indicates that producing the demanded chemical using either technology is advantageous to not producing the CHEM3 (i.e. purchasing the demanded amount). After the first time period, the capacity expansions result in the realization of whether or not each technology successfully completed pilot-plant stage. There are four different outcomes: both TECH1 and TECH2 fail, only TECH1 fails, only TECH2 fails, or both TECH1 and TECH2 successfully complete pilot-plant stage. In time periods two and three, the MILP solution only recommends capacity expansions in the scenarios where both TECH1 and TECH2 successfully complete pilot-plant stages. At time period two, the levels of expansion for TECH1 depends on the realized yield value in each scenario and ranges between 5.7 Mtonnes and 5.9 Mtonnes, whereas, the level of expansion for TECH2 is 5.9 Mtonnes. At time period three, there is expansion in both technologies. The level of capacity expansion is 3 Mtonnes for both technologies. The scenarios where both technologies do not successfully complete pilot-plant stage represent a small portion of the objective function value caused by their probability of occurrence. A tighter optimality gap should result in capacity expansions in scenarios where only one of the technologies successfully complete pilot-plant stage. As with Case Study 1, the solution prioritizes the production of CHEM3 using installed capacity rather than purchasing it.

The MILP of Case Study 3 has 811,008 binary variables, 1.682 million continuous variables, and 13.312 million constraints. A solution for Case Study 3 was obtained in 138 CPU s with an optimality gap of one percent. Similar to Case Studies 2 and 2, we generated a NLP by fixing the integer variable values to the MILP solution. This resulted in an NLP with 3.866 million continuous variables and 7.963 million constraints. Similar to Case Study 2 we were unable to find a solution for the NLP using IPOPT Version 3.12. IPOPT encountered memory errors in the linear equation solver. Case Study 3 is both larger and represents a different type of problem than both Case Studies 1 and 2. In Case Study 3, there is already sufficient capacity to produce CHEM4. The problem considers whether or not it is financially advantageous to introduce a new technology which may replace an existing one. The MILP solution suggested the cheapest way to produce CHEM4 is to use the existing technologies to produce. In every scenario, there is no investment in either of under-development technologies. Unlike Case Studies 1 and 2, there is sufficient installed capacity to satisfy the demand for CHEM4 at all time periods, and the demand is satisfied by production either through TECH1 or TECH3 from CHEM1.

The size of the problem (i.e. the number of variables and number of constraints) is largely impacted by the number of uncertain parameters. The number of uncertain parameters directly impacts the number of scenarios and the scenarios cause the non-linear scaling of MSSPs. The number of scenarios grow exponentially with additional uncertain parameters. This causes exponential growth in the number of variables. It also causes super-linear growth in the number of non-anticipativity constraints. Thus, it is important to understand how the number of uncertain parameters can grow in the NTIP problem. For each additional undeveloped technology, there are four additional uncertain parameters. Two representing the elasticities α and β, one representing the success of technology development, and one representing the yield. When each parameter has two outcomes (except the success of technology development which always has three outcomes), the scenario set has a 24 factor increase in the number of scenarios with the addition of each under-development technology. Increasing the number of possible outcomes also increases the number of scenarios although not exponentially. Similarly, increasing the number of time periods increases the number of uncertain demand parameters. For each additional time period, the scenario set is increased a factor equal to the number of realizations for the demand parameter. The nature of the scaling of the problem is observed in both Case Studies 2 and 3. For instance, Case Study 2 has a 24 factor increase in the number of scenarios. This increase results in an additional 330,000 variables and more than 1.5 million additional constraints. With the increase in problem size, it was possible to solve the tight MILP relaxation of the problem, however the NLP became computationally intractable. Similar scaling is observed when comparing the Case Study 3 to Case Study 1.

3.2.4.3 A Biomass to Commodity Chemicals (BTCC) Case Study

The BTCC case study (Fahmi et al. (2014)) considers a segment of the CPI where the desired product is ethylene (Fig. 3.11). Ethylene can be produced from naphtha using existing cracking technology and from biomass using two routes, Fermentation-Catalytic Dehydration or Gasification-Catalytic Conversion-Catalytic Dehydration. We assume that Fermentation, which produces ethanol from biomass, and Catalytic Dehydration, which converts ethanol to ethylene, are mature technologies, and that Gasification, which produced syngas from biomass, and Catalytic Conversion, which converts syngas to ethanol, are under-development technologies.

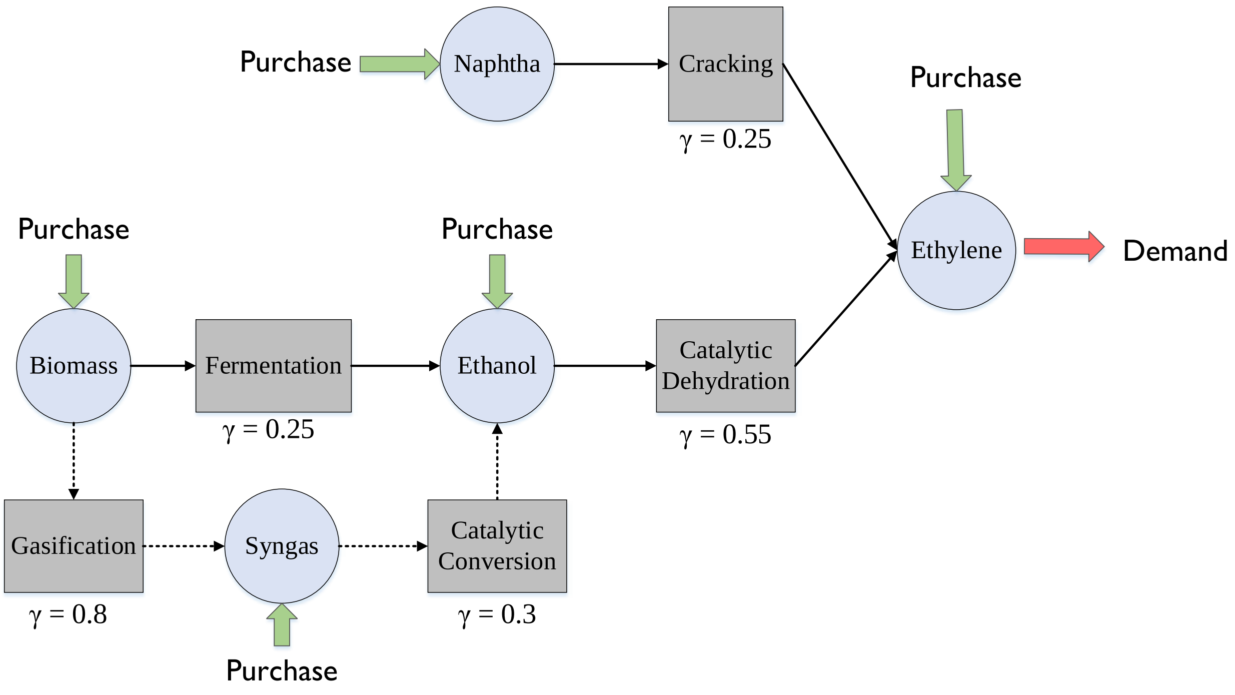


Figure 3.11 The state-task network for the ethylene production case study

The planning horizon is three years separated into one-year time periods. The demand for ethylene at time periods two and three is not known with certainty. Values for the problem parameters and values for the realizations of uncertain parameters are compiled in Tables B.11-B.15 in Appendix B. The MSSP of the BTCC case study has 10 uncertain parameters, including two demand parameters for time periods two and three, and the four uncertain parameters for each under-development technology representing the elasticity parameters (α and β), the yield (χ), and whether or not the technology successfully completes laboratory and pilot-plant stages (ψ). This results in an MSSP with a total of $3^2 \cdot 2^8 = 2304$ scenarios.

The deterministic equivalent of the MSSP, the MINLP model, for the BTCC problem has 1.983 million variables, of which are 89,856 binary variables, and 3.905 million constraints. Baron Version 17.1 was unable to complete the pre-solve step of the problem in the allotted 10,000 CPU s and as a consequence was unable to generate a bound on the MINLP. The MILP (tight relaxation of the MINLP) resulted in 326,241 binary variables and 751,105 continuous variables, and 6.641 million constraints. CPLEX Version 12.63 solved the MILP to an optimality gap of 1% in 2313 CPU s. Using the MILP solution we developed an NLP by fixing the binary variables in the MINLP to the values of the solution of the MILP. We attempted to solve the NLP using IPOPT Version 3.12, however IPOPT encountered an 'out of memory error' caused by the linear equation solver MA27. Increasing the memory allocated to the linear equation solver did not change the outcome, similar to Case Studies 2 and 3.

The objective function value of the MILP solution for BTCC Case Study is \$161 billion. Figure 3.12 plots the empirical CDF (ECDF) for the total cost.

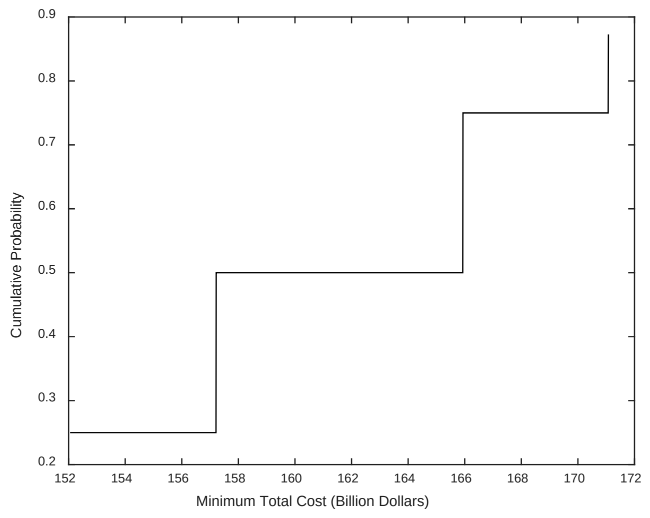


Figure 3.12 The empirical cumulative distribution function for the total cost of the MILP solution

Notice that the total cost of production ranges between \$152 billion and \$172 billion (Figure 3.12). The MILP solution does not invest in either research or capacity expansion for any of the under-development technologies under any scenario. Instead the demand for ethylene is satisfied by using the installed capacity of naphtha cracking. If there is not sufficient capacity to meet the demand through cracking, the remainder of the ethylene is purchased. Given the current prices, a solution favoring the cracking route is consistent with what one would expect to see. It is widely accepted that using biomass for the production of commodity chemicals is not financially advantageous at current market conditions (Uytvanck et al., 2014). The ECDF shown in Fig. 3.12 has four steps, which corresponds to cost differences in the realizations of the demand uncertainty. As there is no capacity or R&D investments to under-development technologies, the values of the uncertain parameters of these technologies are not realized throughout the planning horizon, and that there is only uncertainty realization associated with the demand. An interesting aspect of the optimal solution is lack of investments on capacity expansion of naphtha cracking to meet the demand of ethylene. Instead the solution suggests that existing capacity be used to produce ethylene and any demand that could not be met should be purchased. This indicates that the opportunity cost of producing ethylene versus purchasing ethylene was not high enough to outweigh cost of installing additional cracking capacity.

3.2.5 Conclusion

In this work we have presented a multistage stochastic programming (MSSP) formulation for the evaluation of new technology development projects. The formulation considers uncertainty in the success of developing a new technology, in the demand, and in the cost of installing new capacity. Uncertainty in the problem is modelled using both endogenous and exogenous parameters. The cost of capacity expansion is modelled using a two factor learning curve where the elasticities are treated as endogenous uncertain parameters. The deterministic equivalent of the MSSP formulation yielded a large-scale mixed integer nonlinear programming

(MINLP) model. Non-linearities in the model were linearized using tightly-segmented linear relaxations (Misener et al., 2011), exact linearization (Muhittin and Ossama, 1992), and upper and lower estimators (Fahmi and Cremaschi, 2015). Whether or not a technology successfully completes laboratory and pilot-plant staged and reaches commercial stage was modelled using a stage gate representation. The outcome of the development of the technology and the final yield were modelled as endogenous uncertain parameters. Demand was assumed to be uncertain and was treated as an exogenous uncertain parameter.

This work considered four different case studies. The first case study was a toy-box sized problem which considered the addition of a new technology. The second and third case studies were larger and examined the scaling of the MINLP and MILP models of the NTIP problem. In Case Study 2, two under-development technologies were compared. In Case Study 3, a longer planning horizon retrofitting problem was examined. In both cases, only the MILP models were solved to optimality, and, due to the size of the problem, it was not possible to solve the MINLP and NLP models (using BARON version for MINLP and IPOPT version for NLP) to optimality in 10,000 CPU s. The final case study considered biomass to ethylene production for supplementing or replacing ethylene production via cracking of naphtha. The problem considered uncertainty in the development of the gasification of biomass to produce syngas and uncertainty in the conversion of syngas to ethanol. The solution of the MILP model suggested that, at the current prices, biomass is not a viable investment for the production of ethylene, which is consistent with investment decisions shown in literature.

The NTIP model provides a flexible approach to find the optimal investment planning strategy under capacity cost and demand uncertainty. The approach can be used to model new technology investments and retrofits. Tuning of the model to account for side products and operating costs can be accomplished simply by adding linear terms to the objective function.

The challenge of solving the NTIP problem is two-fold. On one hand, the models within the NTIP problem are non-linear. The solution of non-linear optimization problems is challenging for even small problems creating a need for specialized approaches/algorithms for solving MINLPs and NLPs of this size. On the other hand, the NTIP problem grows rapidly as the size of the problem increases. For instance, an additional undeveloped technology to the model increases the number of variables by a factor of at least 24 and significantly increases the number of constraints. The scaling of the NTIP problem is not unique, MSSPs in general suffer from the curse of dimensionality. The growth of the number of constraints is particularly effected by endogenous uncertainty. Multistage stochastic programs present a particular challenge to researchers There exist many opportunities to develop algorithmic advances which are capable of addressing the computational complexity of solving real-world sized MSSPs.

CHAPTER 4

Multiple Two-Stage SP decomposition (MTSSP)

4.1 Description of the MTSSP Algorithm

The shrinking horizon MTSSP approach (Figure 4.1) generates and solves a series of two-stage stochastic programs (SPs). A similar approach in Balasubramanian and Grossman (2004) solves a series of two-stage SPs using a shrinking horizon for MSSPs with exogenous uncertainty. The authors considered the short-term scheduling of a multiproduct batch plant taking into account demand uncertainties. They formulated an approximate two-stage stochastic programming model for the entire planning horizon. After solving the model, decisions for the first time period are retained, and a new model is generated for the remaining time periods. Their results revealed that the objective functions values obtained by the approximation approach fell within a few percent of the rigorous MSSP objective value. Our work modifies and extends this approach to accommodate endogenous uncertainty. Furthermore, as will be explained in the remainder of this section, in order to keep the number of NACs and the problem size for the two-stage SPs to a minimum, our approach only implements the NACs associated for the first time period.

At each time period, MTSSP approach generates a new two-stage SP based on the availability of resources. The first one is solved at the beginning of the planning horizon (i.e., at $t = 0$). The two-stage problems are generated by removing all NACs except for the current (i.e., first) time-period. For the problem solved at the root node, this means that all the decisions at $t = 0$ must be the same for all scenarios, however, decisions made at later time periods anticipates the outcomes of each individual scenario. The resource and scheduling constraints are retained from the multistage stochastic programming formulation.

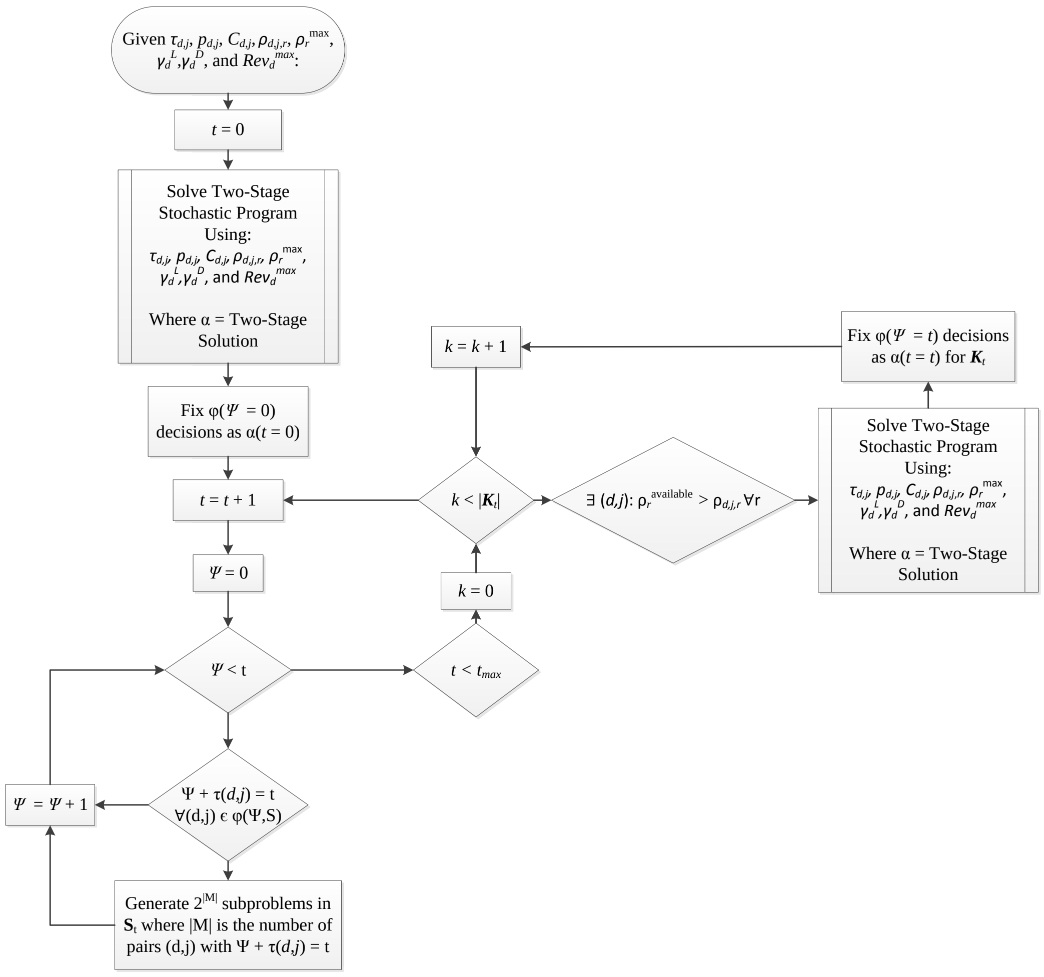

Given $\tau_{d,j}$, $p_{d,j}$, $C_{d,j}$, $\rho_{d,j,r}$, ρ_r^{max}, γ_d^L, γ_d^D, and Rev_d^{max}:
$t = 0$
Solve Two-Stage Stochastic Program Using: $\tau_{d,j}$, $p_{d,j}$, $C_{d,j}$, $\rho_{d,j,r}$, ρ_r^{max}, γ_d^L, γ_d^D, and Rev_d^{max} Where α = Two-Stage Solution
Fix φ(Ψ = 0) decisions as α(t = 0)
$t = t + 1$
$\Psi = 0$
$\Psi < t$
$\Psi + \tau(d,j) = t$ ∀(d,j) ϵ φ(Ψ,S)
$\Psi = \Psi + 1$
Generate $2^{|M|}$ subproblems in $\mathbf{S}_t$ where |M| is the number of pairs (d,j) with $\Psi + \tau(d,j) = t$
$t < t_{max}$
$k = 0$
$k < |\boldsymbol{K}_t|$
$k = k + 1$
∃ (d,j): $\rho_r^{available} > \rho_{d,j,r}$ ∀r
Solve Two-Stage Stochastic Program Using: $\tau_{d,j}$, $p_{d,j}$, $C_{d,j}$, $\rho_{d,j,r}$, ρ_r^{max}, γ_d^L, γ_d^D, and Rev_d^{max} Where α = Two-Stage Solution
Fix φ(Ψ = t) decisions as α(t = t) for $\boldsymbol{K}_t$

Figure 4.1 The MTSSP Algorithm

Additionally, a constraint preventing overscheduling is implemented, *viz.* Eq. 4.1. Eq. 4.1 ensures that any decisions made at the current time period have enough resources to be completed within a time period that is equal to the planning horizon. This equation only takes into account the current decisions, ignoring decisions that were made for the previous time-periods, and considers the resource availability for the overall length of the planning horizon. Using the length of the planning horizon results in less over-scheduling for earlier stages in a drug's development (trial PI) and more over-scheduling as the products progress through the trials (trial PIII). The total necessary resources for trials that are selected for investment, *i.e.*, $X_{d,j,t,s} = 1$, is calculated by multiplying the resource cost [$\rho_{d,j,r}$] of the remaining trials by their durations [$\tau_{d,j}$].

$$\sum_{d}\sum_{j'\geq j}\sum_{j}\tau_{d,j}\rho_{d,j,r}X_{d,j',t,s} \leq |\boldsymbol{T}|\rho_r{}^{max} \qquad \forall r,s \tag{4.1}$$

Eq. 4.2 forces to start at least one trial of a product at the time the two-stage SP is solved. Without this constraint, the solution to the two-stage problem will delay all investments beyond the current time as it lacks the NACs for the remainder of the time-periods.

$$\sum_{i,j} X_{i,j,t_{current},s} \geq 1 \quad \forall s \tag{4.2}$$

Eq. 4.3 replaces all NACs in the original MSSP, and is the set of first time-period NACs. The formulation for the two-stage SPs is obtained by adding Eqs. 4.1-4.3 to Eqs. A.1-A.8 of the MSSP.

$$X_{d,1,1,s} = X_{d,1,1,1} \qquad \forall i,s \tag{4.3}$$

Once the solution to the initial two-stage SP is obtained, only the first-time period decisions are implemented and time is incremented by one (Fig. 4.1). Next, the algorithm checks if there are any resources available for investment and if there are any realizations based on the first-time period decisions at the current time point. If there are resources available and there have not been any realizations, one new two-stage SP is constructed. For this child two-stage SP, the decisions for all scenarios for t < current time are fixed to the first-time period decisions recommended as the solutions of two-stage SP problems solved up to that time point, and the only NACs for the current time point are included. This two-stage SP is solved and time is incremented. On the other hand, if there are resources available and there have been realizations, the number of children two-stage SPs is equal to the number of realized outcomes. The set of scenarios is divided into subsets, each of which contains the scenarios that correspond to one of the realized outcomes. For each scenario subset, a two-stage SP is generated and solved with NACs of the current time. The algorithm continues to march through time until each scenario is realized or until the end of the planning horizon is reached.

Table 4.1 Parameters for the Two-Product Case Study

	Duration		**Probability**		**Cost ($1M)**		**Resource 1 (ρ^{max}=2)**		**Resource 2 (ρ^{max}=3)**		rev^{max}	γ^L	γ^D
product	*PI*	*PII*	*PI*	*PII*	*PI*	*PII*	*PI*	*PII*	*PI*	*PII*			
A	2	4	0.3	0.5	10	90	1	1	1	2	3100	19.2	44
B	2	3	0.4	0.6	10	80	1	2	1	1	3250	19.6	56

Here, we will demonstrate the approach using a two-product two-trial case study of the pharmaceutical R&D pipeline management problem. Parameters for the two product example are shown in Table 4.1. The example consists of two products, *A* and *B*, each of which is required to complete two clinical trials, *PI* and *PII*. The planning horizon is 15-months divided into three-month increments. The rigorous MSSP formulation of the two-product case-study is given by a set of nine scenarios corresponding to the combination of outcomes of each of the two products. The objective function for the MSSP formulation is the ENPV of the decision tree, *viz.* Eq. (A.1). In order to ensure that the decisions do not violate scheduling rules, three binary variables are defined: (1) trial *j* for product *d* is started at time *t* for scenario *s* [$X_{d,j,t,s} = 1$], (2) trial *j* for product *d* is complete at time *t* for scenario *s* [$Y_{d,j,t,s} = 1$], and (3) trial *j* for product *d* is ready to start at time *t* for scenario *s* [$Z_{d,j,t,s} = 1$] (Colvin and Maravelias 2008). The constraints are used to ensure that a trial cannot be started until its predecessor trials are started (Eqs. A.3-A.5 and Eq. A.7), that trials can only start once (Eq. (A.6)), and that a trial is complete after the associated duration (Eq. A.2). Additionally, decisions are constrained by available resources, which are implemented by Eq. A.8. Eqs. A.9 and A.10 are the NACs.

Solving the small two-product case using the rigorous MSSP given in Appendix A yields a solution with an ENPV of $1104 M and the decision tree shown in Figure 4.2a. For the two-product example, the root node SP solution recommends starting trial (*B, PI*) as the first-stage decisions (Fig. 4.2b). After incrementing the time (t = 1), the algorithm determines that there are idle resources, and a new two-stage SP is generated. In this new two-stage SP, the decisions at $t = 0$ are fixed to the first-time period decisions of the initial SP for all scenarios, and the NACs for $t = 1$ are added. For our two-product example, the optimal solution to the two-stage SP at $t = 1$ recommends investing in trial *PI* of product *A*, (*A,PI*) (Fig. 4.2b). At this point, the time is incremented to $t = 2$. At $t = 2$, the algorithm checks to see if there are resources available and if there have been realizations. There are two realizations at $t = 2$, yielding two outcomes: product *A* passes or fails trial *PI*. For this problem, there are a total of nine scenarios, and in three of them, product *A* fails the first trial, *PI*. These three scenarios are grouped as one scenario subset. The other scenario subset includes the remaining six scenarios where product A passes the first trial. Normally, two two-stage SPs would be generated, however, there are not enough resources to start any trials at $t = 2$, and hence, no SPs are generated at this time. Time is incremented to $t = 3$. The algorithm again checks to see if there are resources available for investment and whether or not there are investment decisions available. There are two realizations at $t = 3$. The realizations are: product *B* passes or fails trial *PI*. The scenario sets from $t = 2$ are further divided based the outcome of trial (*B,PI*). This yields four scenario sub-sets and thus four sub-problems. For each of the four sub-problems, the algorithm checks whether there are enough resources or not. For three of the four sub-problems, there are sufficient resources and investment opportunities. For scenario set where both (*A,PI*) and (*B,PI*) fail, there is no investment decision available, i.e., both products failed their first trials. The solutions of the remaining three two-stage problems at $t = 3$ can be seen in Figure 4.2b. Incrementing the time to t=4, we realize that there are no resources available for investment for each child, and we reach the end of the planning horizon. Given this decision tree (Fig. 4.2b), the ENPV of the MTSSP solution is \$1081 M compared to the optimum of \$1104 M.

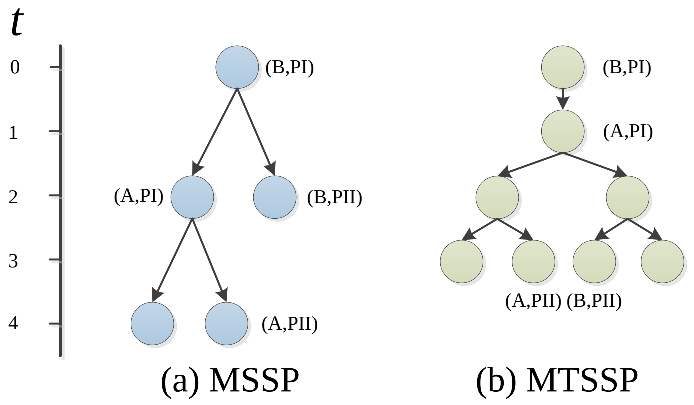


Figure 4.2 Results of the (a) MSSP and (b) MTSSP for the Two-Product Case Study

4.2 Pharmaceutical R&D Pipeline Management Problem Case Studies

In this section, we apply the MTSSP algorithm to solve three instances of the pharmaceutical R&D pipeline management problem. These instances include three, five, and six products. Each should successfully complete three critical trials before reaching market. Parameters for all cases are included in Table 3.3, Table 3.5, and Table 3.6 in Section 3.1.1. Each trial consumes two different resources while running. The MTSSP heuristic was implemented using Pyomo 3.3 (Sandia Corporation, 2013) on 64-bit Ubuntu 12.04. Unless otherwise noted, all programs (including the rigorous MSSPs) were solved to optimality with 0.01% gap using CPLEX 12.51 on IntelCore i7 4770K @ 3.4 GHz x 8 with 16GB RAM. The problems that had extensive memory requirements (more than 16GB RAM) were solved using CPLEX 12.51 on Intel Xeon CPU E5606 @2.13 GHz x 8 with 32GB RAM running 64-bit Ubuntu. The solution times for each method are given in CPU seconds.

4.2.1 Solution to a Three Product Case Study

The percent difference between the objective function obtained with the MTSSP approach ($1169 M) and the rigorous MSSP ($1189 M) solution for the three-product case is 1.6% (Figure 4.3). The to the MTSSP decomposition algorithm was obtained in 25.8 CPU seconds, which is comparable to the solution time of the rigorous MSSP (29.98 CPU seconds).

Total of five two-stage SPs were solved to generate the decision tree shown in Figure 4.4b. Comparison of this decision tree to the one obtained as the MSSP solution shows that the decisions up to t = 3 (Figures 4.4a and 4.4b) are same, and they begin to differ starting at t = 3, where the solution of the rigorous MSSP recommends to wait rather than start trial (*D2,PI*). The current implementation of the MTSSP approach does not allow any "wait-and-see" decisions, and hence at times, leads to over-scheduling. One example of the overscheduling effect in the three product case can be seen at t = 6.

In the MTSSP approach, there are too few resources available to start (*D2,PII*). If (*D1,PI*) is successful this causes penalties to be accessed for D2 being idle in the R&D Pipeline. This tendency to overschedule causes several products to sit idle in the pipeline due to resource unavailability at later times of the planning horizon, and thus yields a lower objective function value.

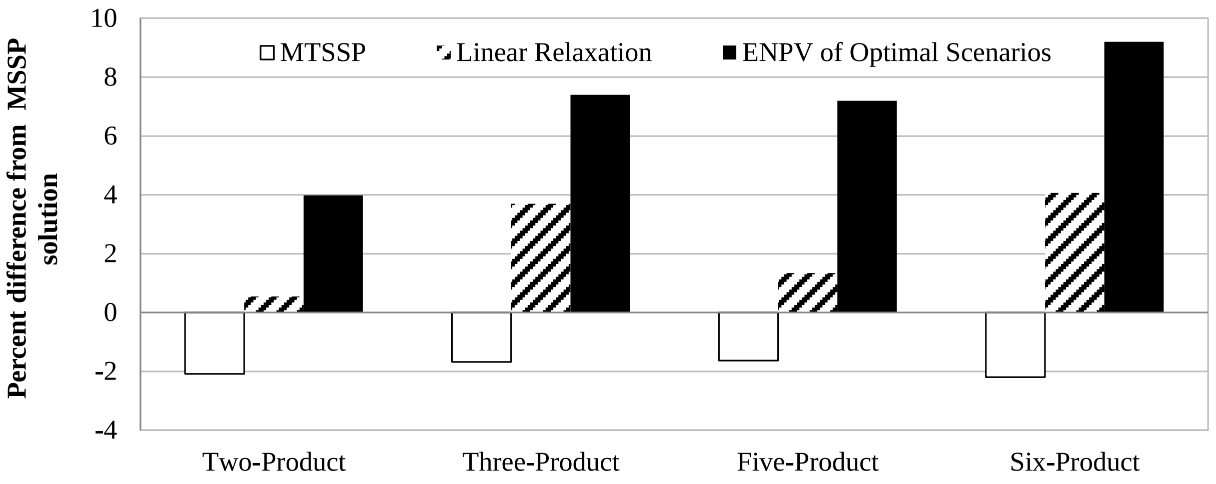


Figure 4.3 Percent differences between the rigorous MSSP solution and the objectives obtained by the MTSSP, Linear Relaxation, and ENPV of Optimal Scenarios

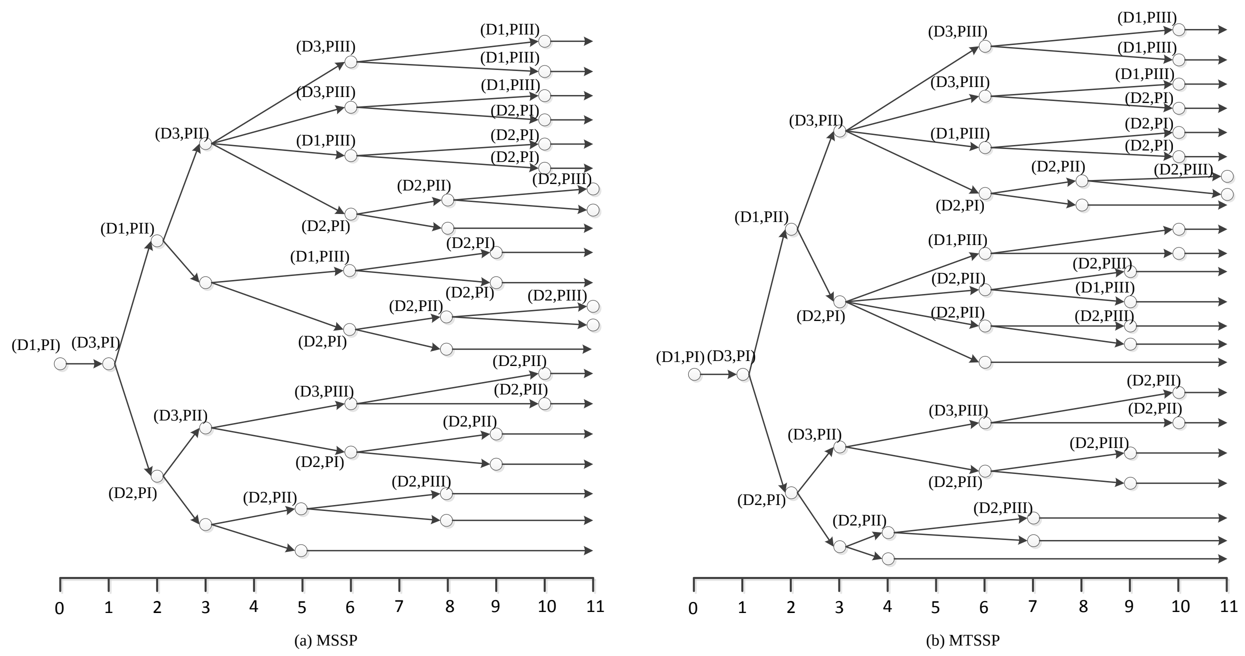


Figure 4.4 Solutions to the three-product case obtained using the (a) the deterministic equivalent MSSP and (b) MTSSP Approach

4.2.2 Application of the MTSSP Algorithm to Larger MSSPs

Other than the two- and three-product problems, the MTSSP approach yielded solutions for five- and six-product problems. Considering all cases, the differences between the ENPVs obtained with the MTSSP algorithm and the ENPVs of the rigorous MSSP solution (the optimum) are all below 3.0 % (Fig. 4.3). The difference between the objective values obtained by the MTSSP approach and the optimal is similar for all instances of the clinical trial planning problem. As the MTSSP approach only considers first-stage decisions at each sub-problem, it makes decisions here-and-now as it marches through the planning horizon, and as such, is unable to consider wait-and-see strategies. This behaviour of the algorithm translates into two limitations while solving multi-stage stochastic programs with endogenous uncertainty: (1) decisions are made based on the highest current NPV, and (2) trials are over-scheduled early in the planning horizon leading to some products sitting idle in the pipeline due to resource limitations that were not anticipated. The MTSSP approach can not anticipate the availability of resources beyond the here-and-now decisions, therefore if the optimal solution were to invest in a lower value product now and a higher value product later in order to exploit resource availability, the MTSSP solution will not capture it. Similar arguments apply for the smaller cases, however this behaviour is often difficult to spot due to shorter planning horizons and fewer products. For medium size cases, such as the three-product case, this behaviour of the approach can be seen at $t = 2$ where the MTSSP solution decides to start (*D1,PI*) rather than wait until the realizations of trial (*D3,PI*) outcome. Making the correct decisions early in the planning horizon had the largest effect on the ENPVs of the solutions. When considering the differences between the MTSSP solution and the MSSP solution for the five-product case, the MSSP root node decision starts the first trials of D3, D4, and D5. The resource requirement to start all of the trials continuously suggests that there will be bottlenecking. The MSSP solution allows for resource overscheduling because the likelihood of all products being successful is low. The MTSSP results for the five-product case also selected items that will exceed the number of available resources if all clinical trials are successful. However, the items selected (D2, D4, and D5) by the MTSSP algorithm had a higher resource cost. Most notably the resources required to start the second trials for the drugs selected by the MTSSP algorithm require four of each type of resources whereas the ones selected by the MSSP solution requires three and four of each type of resource. Decisions beyond the root node for the five-product case are similar in that both will attempt to start the product with the highest expected value that fits within the resource limitations. In the five-product case, if all root node trials fail both remaining products can be started. The solution times for the MTSSP approach were comparable to that of the rigorous MSSP, and were two orders of magnitude faster for the five- and six-product problems (Table 4.2).

Table 4.2 Computational Results for the MTSSP, MSSP, Linear Relaxation, and ENPV of Optimal Scenario Solutions problems

	MTSSP Approach				MSSP		Linear Relaxation		ENPVof Optimal Scenario Solutions ($M)
	Two-Stage Problems Solved	First Problem Time (CPU Sec)	Solver Time (CPU Sec)	Objective ($M)	Solver Time (CPU Sec)	Objective ($M)	Solver Time (CPU Sec)	Objective ($M)	
Two-Product	5	0.05	0.2	1081	0.41	1104	0.07	1110	1148
Three-Product	33	1.4	9.17	1169	29.98	1189	2.95	1233	1277
Five-Product	96	19.37	100.23	2049	6620.35	2083	163.55	2111	2233
Six-Product	235	96.35	478.59	2359	20233.79**	2412	2549	2510	2634

The solution time for the MTSSP approach depends on the number of two-stage SPs that should be solved. This number changes based on the number of products and trials, and the length of the planning horizon. For the worst case scenario, the MTSSP approach will need to solve $2^{|\mathbf{D}||\mathbf{T}|}$ two-stage SPs for a problem with $|\mathbf{D}|$ products that should complete $|\mathbf{J}|$ trials over $|\mathbf{T}|$ time-periods. This estimate assumes that at each stage there are realizations of $|\mathbf{D}|$ products. In our examples, five, 33, 96, 235 two-stage SPs were solved to obtain the solutions for the two-, three-, five-, and six-product problems. Comparing the number of actual problems solved to the theoretical maximum shows that only a fraction of the maximum number of problems are actually solved. Based on our experience with the MTSSP decomposition algorithm, the solution times were typically dominated by the time required to solve the initial two-stage programs as the size of the two-stage SPs diminishes as the planning horizon shrinks. For instance, the initial two-stage problem for the six-product case required 96.35 CPU seconds, whereas the total solution time for the 235 two-stage problems for the six-product case was 478.59 CPU seconds.

The MTSSP approach is easy to implement given an MSSP formulation with endogenous uncertainty and shortens solution times considerably without suffering too much in the solution quality. However, its main limitation is the similarity of its formulation to that of the original MSSP. The two-stage programs of the MTSSP approach have the same number of scenarios as the original problem. Despite the removal of all NACs except the current time-period ones, each of these two-stage programs requires considerable computational memory to generate with the increase in the number of scenarios. Therefore, the MTSSP approach, like the rigorous MSSP formulation, suffers from rapid model growth, and hence, failed to solve the seven- and ten-product instances of the R&D pipeline clinical trial planning problem.

CHAPTER 5
KNAPSACK DECOMPOSITION HEURISTIC (KDA)

5.1 Description of the KDA algorithm

Throughout this section, two-product (toy) example of the pharmaceutical R&D pipeline clinical trial planning problem used in Chapter 4 is used to illustrate the KDA. Penalty coefficients, the duration of each trial, the probability of success for each trial, the maximum revenue, and the cost of each trial are summarized in Table 4.1.

The knapsack decomposition algorithm (Figure 5.1) solves a series of knapsack problems at allowable time periods, which are determined based on the outcomes of uncertainty and availability of resources for investment. The algorithm starts by generating a set of items. The items [$i \in \boldsymbol{I}$] are created by enumerating all possible decisions, i.e., the product-trial pairs. As an example, the two product problem would have four items corresponding to (*A,PI*), (*A,PII*), (*B,PI*), and (*B,PII*). Then, the time is set to zero and the first knapsack problem is generated and solved at the root node. At each time point where a knapsack problem is solved, the algorithm starts by first generating a subset of items [$\boldsymbol{E}_{i,t,k}$] that are eligible to be packed in the knapsack. For instance, at the root node, the two-product example would have two eligible items, (*A,PI*) and (*B,PI*). The other two items would be ineligible because the prerequisite trials have not been completed. Then, the values and the weights of each eligible item, and the maximum knapsack weights are calculated.

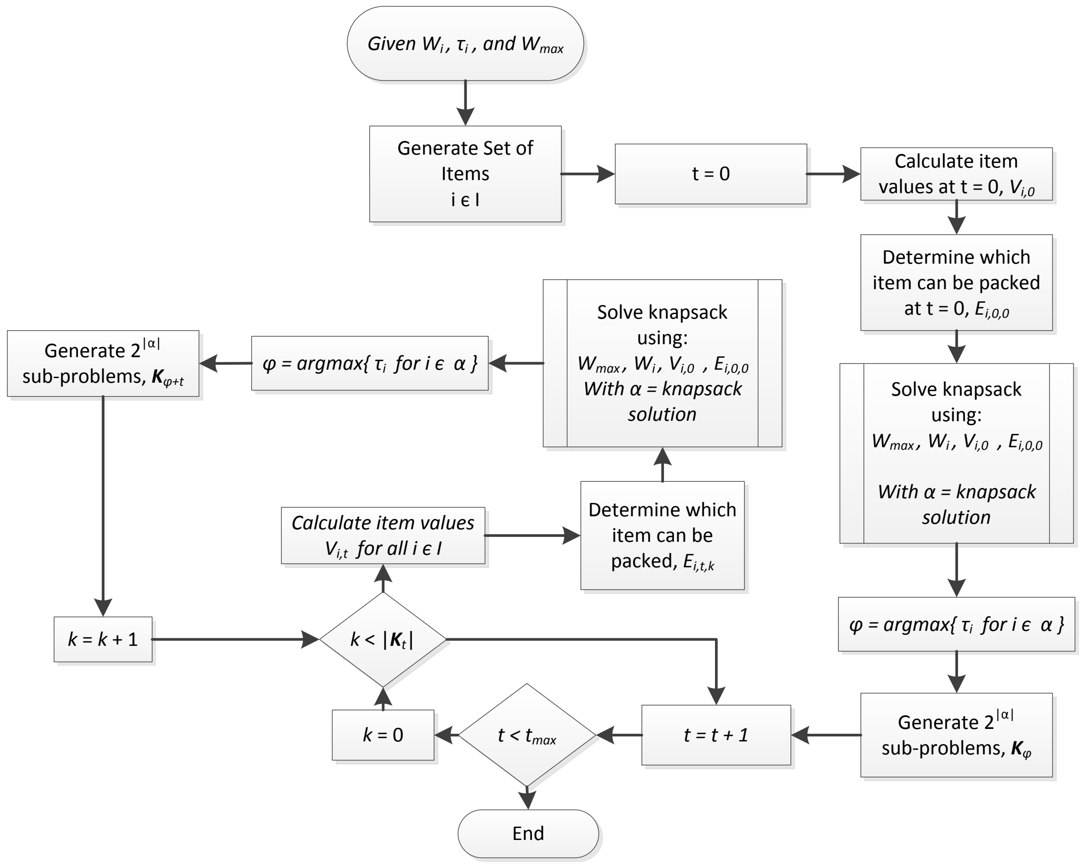

Given W_i, τ_i, and W_{max}
Generate Set of Items i ϵ I
t = 0
Calculate item values at t = 0, $V_{i,0}$
Determine which item can be packed at t = 0, $E_{i,0,0}$
Solve knapsack using: W_{max}, W_i, $V_{i,0}$, $E_{i,0,0}$ With α = knapsack solution
φ = argmax{ τ_i for i ϵ α }
Generate $2^{|\alpha|}$ sub-problems, K_φ
t = t + 1
t < t_{max}
End
k = 0
k < $|K_t|$
Calculate item values $V_{i,t}$ for all i ϵ I
Determine which item can be packed, $E_{i,t,k}$
Solve knapsack using: W_{max}, W_i, $V_{i,0}$, $E_{i,0,0}$ With α = knapsack solution
φ = argmax{ τ_i for i ϵ α }
Generate $2^{|\alpha|}$ sub-problems, $K_{\varphi+t}$
k = k + 1

Figure 5.1 The KDA Algorithm

Each item's value is calculated based on the probability that the product passes the remaining clinical trials and the potential revenue for successfully completing all trials, *viz.* Eq. 5.1.

$$V_{i,t} = \left[Rv_{d(i)} - {\gamma_{d(i)}}^{L} \left(t + \sum_{j' \geq j(i)} \tau_{d(i),j'} + 1\right) \right] \prod_{j \geq j(i)} P\left(P_{d(i),j} = Pass\right) \qquad (5.1)$$

In Eq. 5.1, $Rv_{d(i)}$ represents the potential revenue for the product associated with item i. The potential revenue is calculated by taking the maximum revenue [${Rev_{d(i)}}^{max}$] and deducting the linearly depreciated trial costs [$C_{d(i),j(i)}$] as shown in Eq. (5.2). The potential revenue is depreciated linearly from loss of active patent life [${\gamma_{d(i)}}^{L}$]. The loss of revenue due to products being idle in the pipeline is not included in this formulation because the revenue is calculated assuming that trials are conducted continuously. In Eq. 5.1, the probability of the product d associated with item i passing the remaining clinical trials is used to weigh the depreciated potential revenue.

$$Rv_i = {Rev_{d(i)}}^{max} - \sum_{j' \geq j(i)} C_{d(i),j'} \left[1 - 0.025 \sum_{j'' > j(i)}^{j' \geq j''} \tau_{d(i),j''-1} \right] \tag{5.2}$$

Using Eq. 5.1, the value of each item can be calculated at each time period. For the root node of the toy problem, the values of the eligible items are given as follows:

$$V_{1,0} = [3004.5 - 19.2(0 + (2 + 4) + 1)](0.3)(0.5) = 430.5 \tag{5.3}$$

$$V_{3,0} = [3164 - 19.2(0 + (2 + 3) + 1)](0.4)(0.6) = 731.1 \tag{5.4}$$

Item weights are calculated based on resource requirement(s) for each item. The number of constraints depends on the number of resource types. The maximum capacity of knapsack for each constraint is set to the available resource for investment for each resource type. The capacity constraints for the two product example at the root node are given in Eqs. 5.5 and 5.6.

$$1It_{(A,PI)} + 1It_{(B,PI)} \leq 2 \tag{5.5}$$

$$1It_{(A,PI)} + 1It_{(B,PI)} \leq 3 \tag{5.6}$$

In Eqs. 5.5 and 5.6, It_i is a binary, which is equal to one if item i is packed in the knapsack and zero otherwise. To avoid bottlenecks in the pipeline due to lack of resources, an additional constraint, Eq. 5.7, requires that items packed into the knapsack can complete all trials without idling in the pipeline.

$$\begin{aligned} &\sum_i \left[\sum_{j \geq j(i)} \rho_{d(i),j,r} \tau_{d(i),j} \right] It_i \\ &\qquad \leq \max \left\{ \sum_{j' > j(i)} \tau_{d(i),j'} + 1 \quad \forall i \in \boldsymbol{E} \right\} {\rho_r}^{max} \qquad \forall r \in \mathbf{R} \end{aligned} \tag{5.7}$$

In Eq. 5.7, $\rho_{d(i),j,r}$ is the required amount of resource type r to complete trial j for each product d associated with knapsack item i. The duration of each trial for each product is given by $\omega_{d(i),j}$. For the knapsack problem solved at the root node of the two-product problem, Eq. 5.7 is expressed as:

$$6It_{(A,PI)} + 8It_{(B,PI)} \leq 14 \tag{5.8}$$

$$10It_{(A,PI)} + 5It_{(B,PI)} \leq 21 \tag{5.9}$$

The objective of the knapsack problem is to maximize the value of the packed items. The objective of the root node knapsack problem for the two product case is given in Eq. 5.10. Eqns. 5.5, 5.6, 5.8, 5.9, and 5.10 give the Knapsack problem solved at the root node of the two-product problem.

$$\textbf{Max} \quad (1)(450.6)It_{(A,PI)} + (1)(756.5)It_{(B,PI)} \tag{5.10}$$

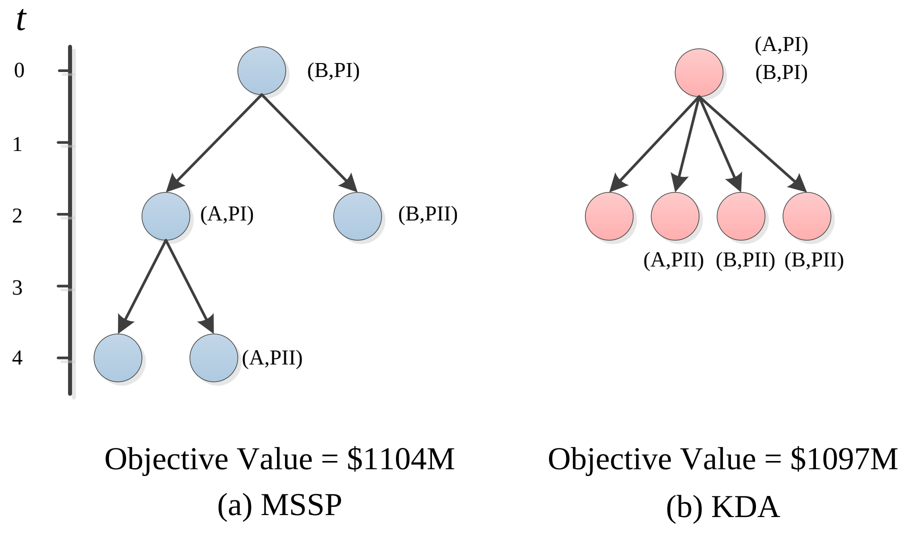


Figure 5.2 The MSSP and KDA results for the two product case study

The optimal solution for the root node selects items (*A,PI*) and (*B,PI*), which recommends investing in the first trials of both products (Figure 5.2b). Once the solution for the root node is found, sets of smaller knapsack problems are generated based on the outcomes of uncertainty associated with the items packed in the previous knapsack. Based on the solution, there are 2^n outcomes, and hence, children nodes, each containing a knapsack sub-problem, where n is the number of items that were packed into the previous knapsack. For the two product case, the root node solution packed two item meaning four additional knapsack sub-problems ($\mathbf{K}_{2,0}$) are generated. These knapsack problems represent the four possible outcomes of trial (*B,PI*) and trial (*A,PI*): both products successfully complete trial *PI*, both products fails at trail *PI*, trail (*B,PI*) is successful and trial (*A,PI*) is failed, and trial (*B,PI*) is failed and trial (*A,PI*) is successful. In the current version of the algorithm, we assume that the sibling Knapsack problems are solved after all outcomes associated with the items that were packed in the parent knapsack are realized. Since more than one item was packed in the knapsack, the subsequent knapsack problems are solved at $t=t+\phi$ whereϕ = Argmax{τ_i $\forall i \in$current knapsack}, *i.e.*, t = 2. The knapsack decomposition algorithm continues to generate children knapsack problems and solving them until the end of the planning horizon is reached or until all products have been pushed through the pipeline and the outcomes of the trials are realized.

For the two-product example, at the second level, one knapsack problem considers the items $\mathbf{E}_{2,1}$= {(*A,PII*), (*B,PII*)}, which is obtained when both products successfully complete trial *PI*. The other sub-problems represent the cases where product *B* fails trial *PI* and product *A* passes trial *PI*, product *A* fails trial *PI* and product *B* passes trial *PI*, and both trials are failed. The subset of items that can be packed in each of these cases is given as $\mathbf{E}_{2,2}$= {(*A,PII*)}, $\mathbf{E}_{2,3}$= {(*B,PII*)},and $\mathbf{E}_{2,4}$= {}, respectively. The solutions to each of the child problems are shown in Figure 5.2b. In the case where $\neq \emptyset$, decisions are to continue investing in products that successfully completed trial *PI*. For the realization that both products pass trail *PI*, the optimal solution to the child problem is to invest in (*B, PII*). The sub-problem where $\boldsymbol{E} = \emptyset$ corresponds to the scenario realization that both products fail their first trial. As the eligible item set is empty, a knapsack problem is not solved for this realization. Repeating the process of determining ϕ for each sub-problem, we realize that the time (t) in which new child problems would be generated lies outside the planning horizon, and hence, the KDA terminates. In other words, for two-product example, the KDA terminates after making decisions at t = 2. The decision tree recommended by the KDA is shown in Figure 5.2b. The ENPV of the KDA solution is \$1097 M, which is within 0.7% percent of the rigorous MSSP solution.

5.2 Pharmaceutical R&D Pipeline Management Problem Case Study

5.2.1 Solution to a three product case study

The KDA was used to solve a three product case considering a 36-month planning horizon divided into 12 three-month increments. The decision trees obtained by the KDA, as well as, the decision tree obtained as the solution of the rigorous MSSP can be seen in Figure 5.3. The objective values and the solution times are shown in Table 5.1. The optimum ENPV is obtained at \$1189 M with a solution time of 30.0 CPU seconds. The optimal decision tree is given in Figure 5.3.

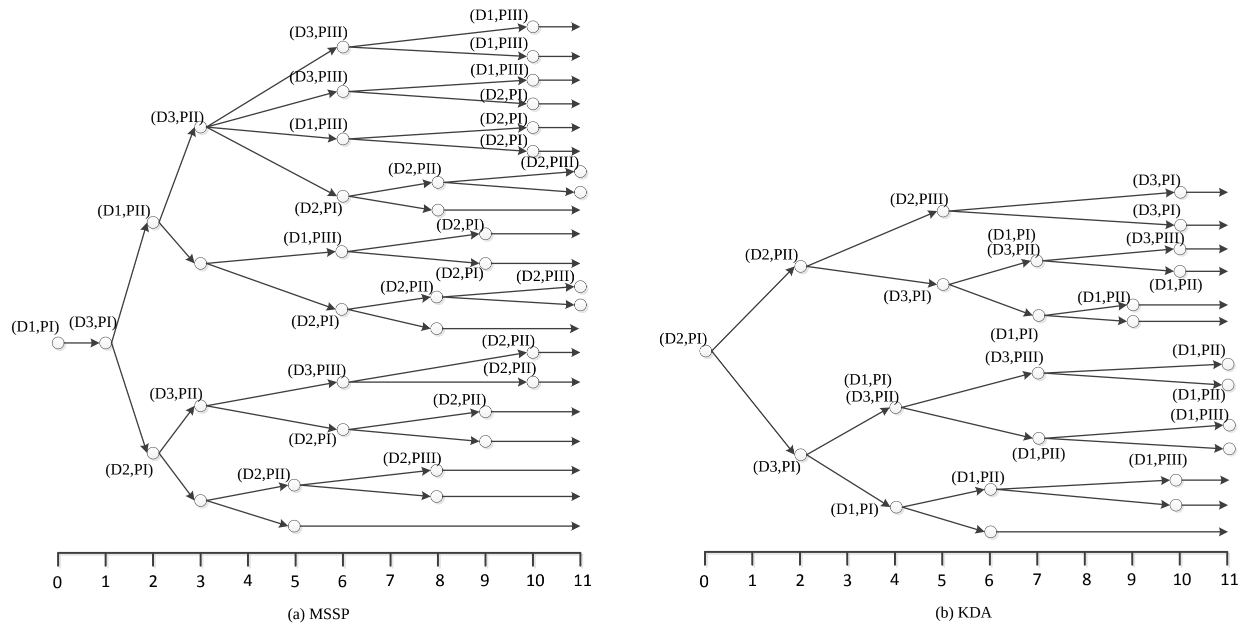


Figure 5.3 Solutions to the three-product case obtained using the (a) the deterministic equivalent MSSP and (b) KDA Approach

For the three-product problem, the KDA produces a decision tree yielding an ENPV of \$1178 M, which is within one percent of the optimal. The KDA solution takes 0.52 CPU second to obtain, which is two orders of magnitude faster than the rigorous

MSSP solution time. The KDA solved a total of 44 knapsack problems, largest of which contained two items, and the smallest one item. Although the difference in ENPVs is small, there are differences between the two decision trees (Figure 5.3a and b). Most notably, the rigorous MSSP solution at t = 0 recommends starting trial (*D1,PI*), whereas the KDA recommends starting trial (*D2, PII*) initially (Figure 5.3a and b). The KDA chooses the "best value" item where the items' values are calculated based on the likelihood of success and the return when the product reaches market. In its current implementation, after a set of decisions are identified (a knapsack problem is solved) for the current time period, the KDA does not generate a new Knapsack problem until the uncertainty regarding all decisions are realized. In other words, the KDA waits to make new investments until all currently invested products complete their trials. This implementation of the KDA may result in underutilization of resources. For instance in the three-product case, the root node decision for the KDA is to start (*D2,PI*). The next decision point for the KDA occurs at t = 2. In the MSSP solution, the root node decision is to select (*D1,P1*). Since the MSSP is not limited to making decisions after all realizations have occurred, the approach chooses to start (*D3,P1*) at t = 1. This difference causes the KDA solution to be sparser than both the MTSSP solution and the MSSP solution. The sparsity in the decision tree of the KDA solution (comparison of Figure 5.3a and b) suggests that the decision to wait until all uncertainties related to the selected items are realized may be too limiting.

5.2.2 Application of the KDA to Larger Pharmaceutical R&D Pipeline Management Problem

The KDA was able to solve all of case studies very quickly (less than 1500 CPU seconds). The solutions obtained using the KDA were within three percent of the solution of the rigorous MSSP (Figure 5.4). In the six-product case, there is less than one percent difference between the CPLEX 12.51 solution of the rigorous MSSP and the KDA solution. However, it must be noted that the CPLEX optimality gap for the six-product case was set to five percent.

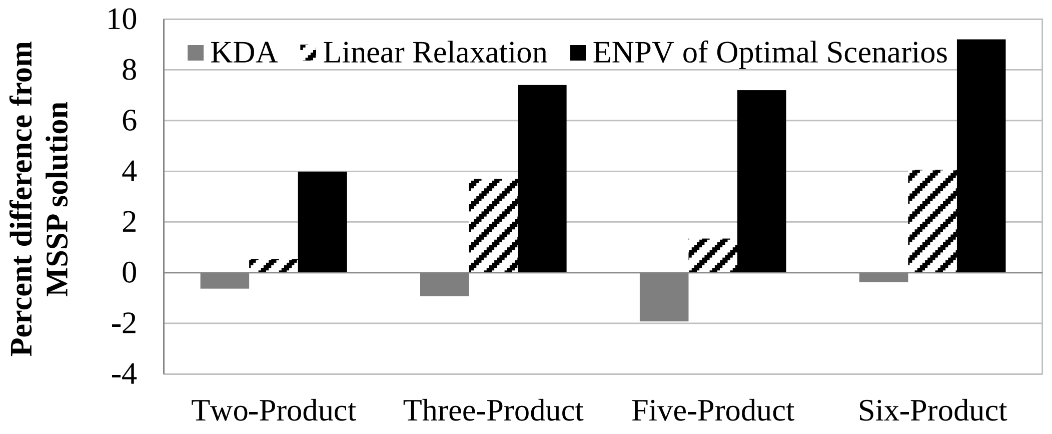


Figure 5.4 Percent differences between the rigorous MSSP solution and the objectives obtained by the KDA, Linear Relaxation, and ENPV of Optimal Scenarios

When the KDA solutions are compared to the actual MSSP solution, it can be seen that the solutions deteriorate as the problem size and planning horizon increase (Table 5.1). A close examination of the three-product-case decision tree revealed that the solutions obtained using the KDA tended to be sparser than the rigorous MSSP algorithm. The decision to wait until all selected items have completed their trials before generating new knapsack problems may have partially contributed to the deterioration of the solution, although some deterioration is expected with the increase in problem size. When considering the decision tree for the five-product case the sparsity of the KDA solution is less noticeable than with the three-product case. In the three- product case, the durations of the trials range from two to four time periods whereas with the five-product case the durations of clinical trials range between one and three time periods. With the durations of the trials being shorter realizations occur more frequently allowing for more products to be scheduled. For instance, in the three-product case, the KDA was required to wait until $t = 2$ to start new clinical trials, however in the five-product case, the realization of root node decisions occurs at $t = 1$. Additionally, the KDA solution for the five-product case selects fewer items than the MSSP for the root node problem.

Table 5.1 Computational results for the KDA, MSSP, Linear Relaxation, and ENPV of Optimal Scenarios problems

	KDA Approach				Deterministic Equivalent MSSP		Linear Relaxation		ENPVof Optimal Scenario Solutions
	Knapsacks Solved	Algorithm Time (CPU Sec)	Solution Evaluation (CPU Sec)	Objective ($M)	Solver Time (CPU Sec)	Objective ($M)	Solver Time (CPU Sec)	Objective ($M)	Objective ($M)
Two-Product	4	0.06	0	1097	0.41	1104	0.07	1110	1148
Three-Product	44	0.34	0.03	1178	29.98	1189	2.95	1233	1277
Five-Product	96	0.76	0.36	2043	6620.35	2083	163.55	2111	2233
Six-Product	109	1.09	1.82	2403	20233.79**	2412	2549	2510	2634
Seven-Product	496	5.25	12.52	2870	---	---			3107
Ten-Product	3236	298.35	1462.01	4012	---	---	--	--	5026

The KDA solution for the four product case study selects the first trials for products D2 and D4 at the first time period. (*D2,PI*) and (*D4,PI*) correspond to the items with the highest expected value. The resource overscheduling constraint prevents the KDA from starting clinical trials if the subsequent trials will be resource constrained. The MSSP solution also selects three products at the root node however the MSSP considers resource availability beyond the root node decision point. In the five-product case, the MSSP selects the first trials of products D3, D4, and D5. The solution overschedules resources for the five product case because the likelihood of a failure exceeds the potential loss associated with products that are idle in the pipeline. This result yields the solution that has the highest expected net present value. Decisions that occur beyond the root node decision point favor high value items that are within the available number of resources. In the five product case, if all products selected as the root node decisions fail their initial clinical trials the remaining items may all be started, and this is reflected in the solution. It should be noted that the KDA was the only approach that generated and implementable (i.e., feasible) solutions for problems where the rigorous MSSP cannot be solved.

For seven- and ten- product cases, for which we were not able to obtain the optimum as the solution of the deterministic equivalent of the MSSP formulation due to computational memory constraints, an upper bound of the optimal was calculated using the ENPV of the optimal solutions of each individual scenario. The solution generated by the KDA is within 25% of the solution obtained by the Expected value of the Expected Value of Perfect Information (EVPI) approach for the ten-product case. When the optimum solution for the three-, five-, and six-product cases are compared to the ENPV obtained using the EVPI approach, it is observed that they are within 9.2 % of the MSSP solution. Based on these results, we can argue that for the ten-product case, the solution of the KDA is, at the worst case, within 15% of the optimal MSSP solution.

Unlike the MSSP, the KDA does not require using a full set of scenarios to generate the problem, which considerably reduces the number of variables. This reduction translates into substantially smaller optimization problems. In the six-product case, the rigorous MSSP had 1,359,873 variables. In contrast, the largest knapsack problem the KDA approach solved had 19 variables.

The overall solution time for the KDA is impacted by two steps: (1) executing the KDA, and (2) generating the corresponding decision tree and calculating the ENPV of this decision tree. The CPU seconds required to complete each step for all cases are given in Table 5.1. As can be seen from Table 5.1, the overall solution times for the KDA are dominated by the second step, generation of decision tree and evaluation of the corresponding ENPV for the seven- and ten-product cases. For instance, the time required to execute the algorithm was 298.35 CPU seconds, and the time to generate the ENPV was 1462.01 seconds in the ten-product case. The time to generate the decision tree and evaluate the ENPV grows exponentially due to the exponential growth in the number of scenarios. For smaller instances of the KDA, instances with few scenarios, the times to generate the corresponding decision tree and to evaluate the ENPV is almost negligible when compare to the time to complete the KDA. Nevertheless, the total solution times for KDA are several orders of magnitude lower than the ones for the rigorous MSSP.

The execution times for the KDA depend on the number of knapsack problems solved. A theoretical bound on the maximum number of knapsacks generated can be given by Eq. 5.11.

$$Max\ Knapsacks = 2^{\frac{|\boldsymbol{T}|\xi}{\min\{\tau_{d,j}\ \forall d,j\}}} \tag{5.11}$$

ξ is given by either the solution to the fractional knapsack problem assuming that all of the items can be packed or the number of products. The minimum of these two values is selected. Using the number in Eq. 5.11 yields a very high bound on the number of knapsacks that can possibly be solved. The cause of the high bound is two-fold, (1) the assumption that realizations occur as often as the minimum trial duration, and (2) the assumption that the knapsack always packs the maximum number of products. For most cases considered in this work, the durations of trials vary significantly. For instance, in the three-product case, the duration of the phase III clinical trials is at least twice as long as the phase I clinical trials. The result is an over estimate of the total number of realizations. Additionally, in our experience the number of items packed in each knapsack rarely reaches the maximum number that can be packed. Eq. 5.11 does provide a maximum bound despite the actual number of knapsacks solved being only a fraction of the maximum.

Compared to the bound, the number of knapsacks that needed to be solved when implementing the KDA was much smaller. In two-product case, the case with the highest percentage of the total knapsack problems solved, four knapsack problems were solved. These knapsack problems account for 12.5% of the total possible number of knapsack problems. For cases with more than three-products, solving many smaller knapsack problems is orders of magnitude faster than solving the rigorous MSSP. The KDA provided the quickest solutions with small deterioration in the solution quality when compared to the MTSSP approach presented in Chapter 4..

5.3 Evaluating Sensitivity of the KDA Solution Quality and Time to Original Problem Parameter Values and Size

The parameters of the pharmaceutical R&D pipeline management problem are the lengths and the costs of the clinical trials, the revenue realized after successful completion of all clinical trials, and the penalty factors associated with delay of products already in the pipeline and loss of patent life. We test the sensitivity of the

KDA solution to the values of these parameters by perturbing each one individually. The size of the R&D pipeline management problems changes with the number of trials, the length of the planning horizon, and the number of resources. To study the impact of problem size on the KDA performance, we constructed a set of problems where the number of products, the number of trials, the length of the planning horizon, and the number of resources are varied independently.

5.4 Variants of KDA Decision Rules

The KDA has two decision rules. The first one determines when new knapsack problems are generated. The previous computational studies revealed that although the KDA yielded tight feasible solutions (within three percent), for some problems, particularly the ones with large differences in clinical trial lengths, it generated very sparse decision trees. For these problems, the quality of the solution was worse (closer to three percent). We hypothesize that the sparse decision tree may be a result of the algorithm requiring that all started trials must be completed prior to starting new trials.

The second decision rule specifies the constraint that aims to prevent products from being idle in the pipeline. We refer to this constraint as the resource overscheduling constraint. The original KDA formulation introduces a hard resource overscheduling constraint, which ensures that for every item packed there will be sufficient number of resources to continue subsequent trials. Because this constraint does not consider the possibility that a product may fail a trial, it may significantly limit future investments in additional products, and hence, may lead to sparse decisions trees.

5.4.1 Sub-problem generation rules

Here, two additional approaches are proposed for determining when knapsack problems are generated: (1) at Each Time Period (ETP), and (2) After Each Realization (AER). The ETP generates knapsack problems at each time period where there are idle resources and clinical trials that can be started. When a realization occurs, i.e., one or more of the started clinical trials are completed, new knapsack problems are generated for each realized value. If any items remain in the knapsack (trials that have been started but not yet completed), they are passed as already selected items to each newly generated knapsack problem. Figure 5.5 graphically depicts the knapsack generation schemes using ETP (Fig. 5.5(a)), the original KDA (Fig. 5.5(b)), and AER (Fig. 5.5(c)) for three time periods. The solution of the first knapsack problem is to pack items 1 and 2 (Figure 5.5, $t = 0$). The trial associated with item 1 is completed at $t = 1$, while the trial of item 2 is completed at $t = 3$. The ETP approach generates two new knapsack problems at $t = 1$ (Fig. 5.5(a)). Each knapsack problem corresponds to a unique realization associated with item 1. The binary associated with item 2 is set equal to one ($x_2 = 1$) in both knapsack problems. For this example, we assume that there are not sufficient resources to add any more items at $t = 1$. At $t = 2$, there are no realizations, i.e., the trial associated with item 2 is not completed. The ETP algorithm generates and solves two new knapsack problems. In each knapsack problem, the value of x_2 is set equal to one. The solutions of these knapsack problems are different. In one case, the algorithm selects item 6. In the other case, the algorithm does not have sufficient resources to add another item. The uncertainty realization associated with item 2 occurs at $t = 3$. As can be seen in Fig. 5.5(a), the KDA generates two knapsack problems for each branch. In the case where item 6 is selected, the knapsack problem corresponding to one realization is able to add item 5. The other knapsack problem does not have sufficient resources to add another item.

Figure 5.5(b) depicts the knapsack problem generation rule used in the original KDA. The solution at $t = 0$ selects items 1 and 2. The original KDA does not generate any knapsack problems until the clinical trials associated with both items are completed, i.e., until $t = 3$ (Fig. 5.5(b)). At $t = 3$, the original KDA generates and solves four new knapsack problems, each one corresponding to one of the possible outcomes of uncertainty.

The AER approach (Fig. 5.5(c)) generates and solves knapsack problems when any of the selected trials is completed, i.e., the outcome of an uncertain parameter is realized. Unlike the original KDA, if more than one item were packed in the knapsack, the AER approach generates new knapsack problems at the completion of the shortest clinical trial. Similar to ETP approach, if there are any remaining items in the knapsack (trials that have been started but not yet completed), they are passed as already selected items to each newly generated knapsack problem.

Figure 5.5(c) shows that the solution of the first knapsack problem is the selection of items 1 and 2 at $t = 0$. The uncertainty associated with item 1 is realized at $t = 1$. The AER approach generates new knapsack problems for each realization. Similar to the ETP approach, the AER approach finds that there are insufficient resources to start any new trials. Unlike the ETP approach, the AER approach does not generate new knapsack problems unless there is a realization. Therefore, no knapsack problems is generated at $t = 2$. At $t = 3$, four knapsack problems are generated, two for each branch. The two knapsack problems in each branch represent the possible realizations of uncertainty associated with item 2. Solutions for each of the four knapsack problems at $t = 3$ are shown in Fig. 5.5(c).

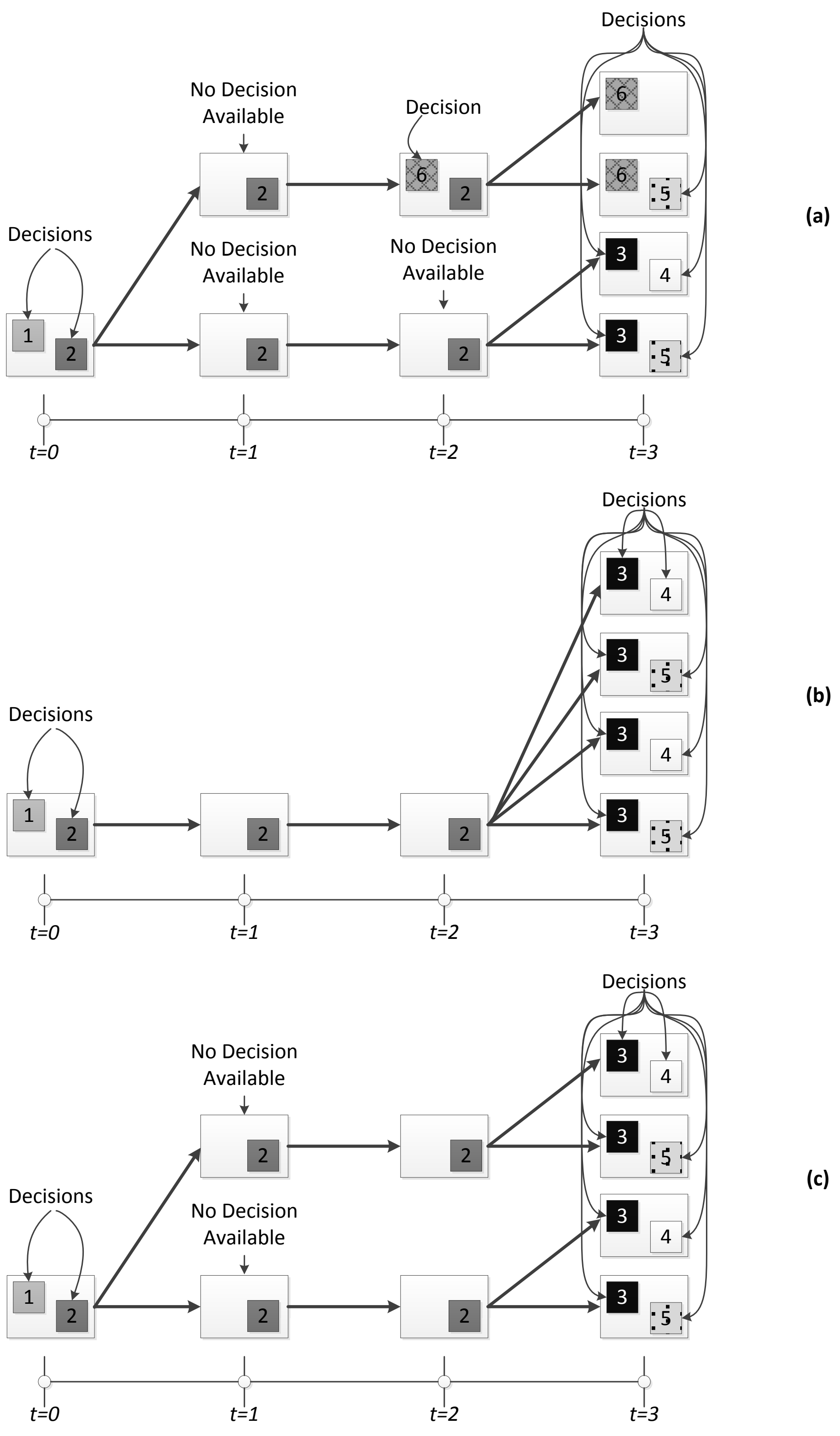


Figure 5.5 The knapsack generation schemes, (a) ETP, (b) Original KDA, (c) AER

5.4.2 Formulations for Resource Overscheduling Constraint

The original resource overscheduling constraint in the KDA does not allow for items (drug-trial pairs) to be packed as part of the solution if there will not be enough resources to pack items corresponding to subsequent trials of the same drug. For instance, assume that two potential drugs need to complete two trials before reaching market, the duration of all trials are equal, and the resource costs are 10 and 30 for drug A trial one and trial two, and 20 for both trials for drug B, and the maximum amount of available resources is 30. The first trials of products A and B cannot be started at the same time because the number of resources needed to start the second trials exceeds the maximum amount. This constraint leads to a conservative solution because it does not consider the possibility of a drug failing to successfully complete the selected trial or the subsequent trials when trying to anticipate the future resource requirements.

We present two new formulations for avoiding possible resource overscheduling. The first one modifies the knapsack problem formulation by adding a penalty term to the objective function rather than an additional constraint. The penalty term grows proportional to the number of resources that exceed the number of available resources, and is shown in Eq. 5.12.

$$P_r = \lambda \sum_{i} \left[\sum_{j \geq j(i)} \rho_{d(i),j,r} \tau_{d(i),j} \right] x_i - \left(\max\left\{ \sum_{j' > j(i)} \tau_{d(i),j'} + 1 \quad \forall i \in \boldsymbol{E} \right\} {\rho_r}^{max} - C_r \right) \tag{5.12}$$

The amount each resource in excess affects the penalty, P_r, is given by a rate constant λ. Penalties are only incurred if the number of resources used exceeds the number of available resources. The number of resources that exceed the number of available resources is calculated by subtracting the maximum resources available $\left(\max\left\{\sum_{j'>j(i)} \tau_{d(i),j'} + 1 \quad \forall i \in \boldsymbol{E}\right\}{\rho_r}^{max}\right)$ from the resources allocated for items that have been previously packed in the knapsack but have not been completed, C_r, and resources allocated for newly packed items. To enforce that the penalty is only imposed when the number of resources is exceeded, a disjunction is introduced with a binary variable, y_r, which is one when the number of resources used exceeds the number of resources available and zero, otherwise. The disjunction can be seen in Eq. 5.13.

$$\begin{bmatrix} y_r \\ Penalty = P_r \end{bmatrix} \vee \begin{bmatrix} \neg y_r \\ Penalty = 0 \end{bmatrix} \tag{5.13}$$

We use big-M formulation to convert this disjunction. The objective function with the penalty term is given in Eq. 5.14.

$$\max\left\{\sum_i V_i x_i - \sum_r P_r y_r\right\} \tag{5.14}$$

Expanding the penalty term in Eq. 5.14 yields Eq. 5.15, whose first term is non-linear due to the multiplication of the binary variables y_r and x_i.

$$\begin{aligned} \sum_r P_r y_r = \sum_r \lambda \sum_i \left[\sum_{j \geq j(i)} \rho_{d(i),j,r} \tau_{d(i),j}\right] x_i y_r \\ - \left(\max\left\{\sum_{j' > j(i)} \tau_{d(i),j'} + 1 \quad \forall i \in \boldsymbol{E}\right\} \rho_r{}^{max} + C_r\right) y_r \end{aligned} \tag{5.15}$$

This term is linearized by introducing a new binary variable z where $z_{i,r} = It_i \cdot y_r$ and adding the following constraints, Eq. 5.16, to the knapsack problems.

$$\begin{aligned} z_{i,r} &\leq x_i \quad \forall i, r \\ z_{i,r} &\leq y_r \quad \forall i, r \\ z_{i,r} &\geq y_r + x_i - 1 \quad \forall i, r \end{aligned} \tag{5.16}$$

The second formulation replaces the original resource overscheduling constraint with a probabilistic constraint, which uses the probability that resources will be needed. The probability that a drug will require resources for a given trial is equivalent to the probability that the drug passes the previous trial(s). The probability that a drug passes a set of trials can be calculated using the probabilities of success in each individual trial. The probability of a series of events occurring, Pr, where the outcome of each event $(e_1, e_2, \dots, e_N)$ is assumed to be independent, can be written as $Pr = P(e_1 \cup e_2 \cup \ldots e_N) = \prod_n P(e_N)$. Assuming the trial outcomes are independent, the probability that a drug will pass a set of trials is calculated as the geometric sum of the probability of each trial's success. Eq. 5.17 shows the modified constraint.

$$\begin{aligned}\sum_i \left[\rho_{d(i),j(i),r} \cdot \tau_{d(i),j(i)} + \sum_{j>j(i)} \left[\prod_{j>j'>j(i)} P(j' = Pass)\right] \rho_{d(i),j,r} \cdot \tau_{d(i),j}\right] \\ \leq \max\left\{\sum_{j'>j(i)} \tau_{d(i),j'} + 1 \qquad \forall i \in \boldsymbol{E}\right\} {\rho_r}^{max} \qquad \forall r \in \mathbf{R}\end{aligned} \tag{5.17}$$

Notice that the new constraint is a relaxation of the original constraint. Resources for the current trial are allocated $(\rho_{d(i),j(i),r} \cdot \tau_{d(i),j(i)})$ but the resources for subsequent trials are weighted using the probability that all of the previous trials are successful $\left(\sum_{j>j(i)}\left[\prod_{j>j'>j(i)} P(j' = Pass)\right]\rho_{d(i),j,r} \cdot \tau_{d(i),j}\right)$.

5.5 Computational Studies

The first set of computational studies investigates the impact of the problem parameters on the KDA performance. The parameters are the trial cost, the revenue for successful completion of the pipeline, the penalties for loss of patent life and non-investment in products currently in the pipeline, the length of each clinical trial, and the overall resource availability. The original two-product (2_2_5_2), three-product (3_3_12_2), four-product (4_3_6_2), and five-product (5_3_6_2) cases are the base case problems. In all but the two-product case, three trials are required to be completed. The two-product case requires the completion of only two clinical trial. Each case has two resources constraining investment decisions. Planning horizons in each base case range from five time periods to 12 time periods. The parameters for each of the base case problems is given in Appendix C. The values of the cost, revenue, and penalty parameters are perturbed by ±10% and ±25% for each case. The sensitivity of the KDA performance to the lengths of the clinical trials is studied by extending the length of each trial by one and two time periods. To study the impact of overall resource availability, four problems are constructed with varying degrees of overall resource constraints: (1) unconstrained, (2) 40 percent unconstrained, (3), 70 percent unconstrained, and (4) fully constrained. The base case problems are assumed to be fully resource constrained. The unconstrained case provides enough resources for each product to enter the pipeline and all eligible clinical trials to be completed simultaneously without delay. The number of resources in the 40 percent unconstrained case is calculated by increasing the available resources of the fully constrained case by 40 percent of the difference between the available resources in the unconstrained and the fully constrained cases. In the 70 percent unconstrained case, this increase is 70%.

The second set of computational studies investigates the effect of the problem size on the KDA solution times. For each of the base case problems, the size of the problem is increased by increasing the number of trials, the length of the planning horizon, and the number of resources. Table 5.2 summarizes the considered variations.

Table 5.2 Problem specifications used for studying the sensitivity of KDA solution time to problem size.

Variation	Variation Value
Number of Trials	+1
	+2
	+3
Number of Resources	+1
	+2
	+3
Length of Planning Horizon	-1
	+1
	+2
	+3
Trial Cost	-25%
	-10%
	+10%
	+25%
Active Patent Life Loss Penalty	-25%
	-10%
	+10%
	+25%
Idle Product Penalty	-25%
	-10%
	+10%
	+25%
Trial Duration	+1
	+2
Percent Unconstrained	0%
	40%
	70%

A plus sign (+) in a Table 5.2 refers to an increase in magnitude, and a minus sign (-) refers to a decrease. The number next to the sign indicates the magnitude of the increase/decrease. A total of 124 problems were developed to test the sensitivity of the KDA to parameter values and problem size. Information for the variations of the specific problems is summarized in Appendix E.

The performance of knapsack generation approaches (ETP and AER) are tested using the six base case problems. The resource overscheduling constraints were analyzed using the original six problems.

Performance of the KDA is evaluated based on its solution quality and computational time requirements. The quality of the solution is assessed by comparing the KDA solution objective function value with that of the rigorous MSSP solution. They are expressed as the percent difference from the rigorous MSSP solution. The computation times are given in CPU seconds. Because the KDA generates solutions orders of magnitude faster compared to the time it takes to obtain the solution for the rigorous MSSP using commercially available solvers, the computation times for the variants of the KDA are compared to that of the original KDA.

The KDA and the deterministic equivalent formulation of the rigorous MSSP have been implemented using python 3.5 with Pyomo 4.1 (Sandia Corporation, 2013) on Auburn Hopper. Pyomo solves each knapsack sub-problem using CPLEX 12.63 to an optimality gap of 0.1%. The rigorous MSSPs are solved to an optimality gap 1% for the two-, three-, and four-product variations and a gap of 5% for the five- and six-product variations. In both cases, the rigorous MSSP is solved using CPLEX 12.63.

5.6 Results and Discussion of Computational Studies

5.6.1 The Impact of Changes in the R&D Pipeline Management Problem Parameters

The computational experiments revealed that the changes in the trial cost(s), the revenue(s), and both penalty parameters had very little effect on the decision trees generated by the KDA. Changes in the parameters, however, (as expected) did have fairly large changes in the value of the objective function. The numerical results of these studies are compiled in Appendix D. Similarly, variations in lengths of clinical trials did not have a significant impact on the solution quality.

Figure 5.6 plots the fraction of the ENPV of the KDA and MSSP for each of the trials vs the ENPV of the fully unconstrained case for each of the base cases. One can observe that the rigorous MSSP ENPVs asymptotically approach the unconstrained solution. A similar behavior is observed for the KDA solutions. However, the KDA solution asymptotically converges to a solution that has a lower ENPV than the MSSP in all but the two-product base case (Fig. 5.6). This behavior is an artifact of the knapsack problem generation schema. In the original KDA, knapsack problems are only generated after all realizations occur. Therefore, unless all clinical trials have the same duration penalties are accessed for products sitting idle in the pipeline, and the resulting ENPV is lower than the ENPV of the rigorous MSSP solution.

As for the impact of the size of the problem on the quality of the KDA solution, when the length of the planning horizon and the number of resources were varied the KDA remained within five percent of the deterministic MSSP solution. However, when the number of trials was increased the KDA produced solutions greater than 30% lower. After examining the decision trees for both the KDA and the deterministic MSSP where the difference in ENPVs exceeded five percent, it revealed that the solution for the MSSP was the "do nothing" solution. Since the KDA only has positive value items (see Eq. 5.2), it is unable to yield the "do nothing" solution.

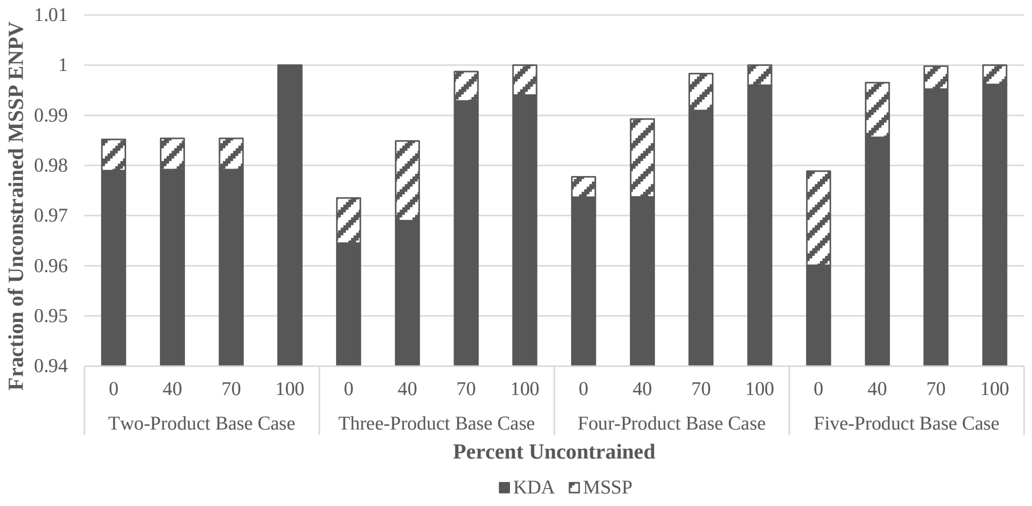


Figure 5.6 Change in ENPV of the KDA and MSSP solutions with resource availability

5.6.2 The Impact of R&D Pipeline Management Problem Size on KDA Solution Time

Figure 5.7 plots how KDA solution times and the number of knapsack problems solved change with number of resources, number of clinical trials, and length of planning horizon. The top row of charts in Figure 5.7 plot the KDA solution times versus the variation of resources, number of clinical trials, or length of planning horizon. The charts plotting KDA solution time reveal that the number of resources and clinical trials have negligible impacts on KDA solution times. The number of resources affects the number of weight constraints in the knapsack problems. Adding weight constraints increases the complexity of an individual knapsack problem, and hence, may increase its solution time. For the level of variation considered in the number of resources for this computational study, the solution times for individual knapsack problems did not change significantly. The number of trials affects the total number of items in knapsack problems, but does not change the number of items that can be packed in any given time. The maximum number of items that may be packed in any knapsack is equal to the number of products.

The KDA solution times appear to grow significantly when the length of the planning horizon is increased. (Fig. 5.7). The increase in planning horizon corresponds to an increase in potential decision points. An increase in decision points increases the number of knapsack problems solved, and hence, the KDA solution time.

The bottom row of charts in Fig. 5.7 plots the number of knapsacks generated versus the variation of the number of resources, number of trials, and length of the planning horizon. One of the trends revealed in these plots is the positive correlation between the KDA solution time and the number of knapsack problems solved. The trend is most noticeable when comparing the solution time and the number of knapsack problems solved when the variation is in the length of the planning horizon.

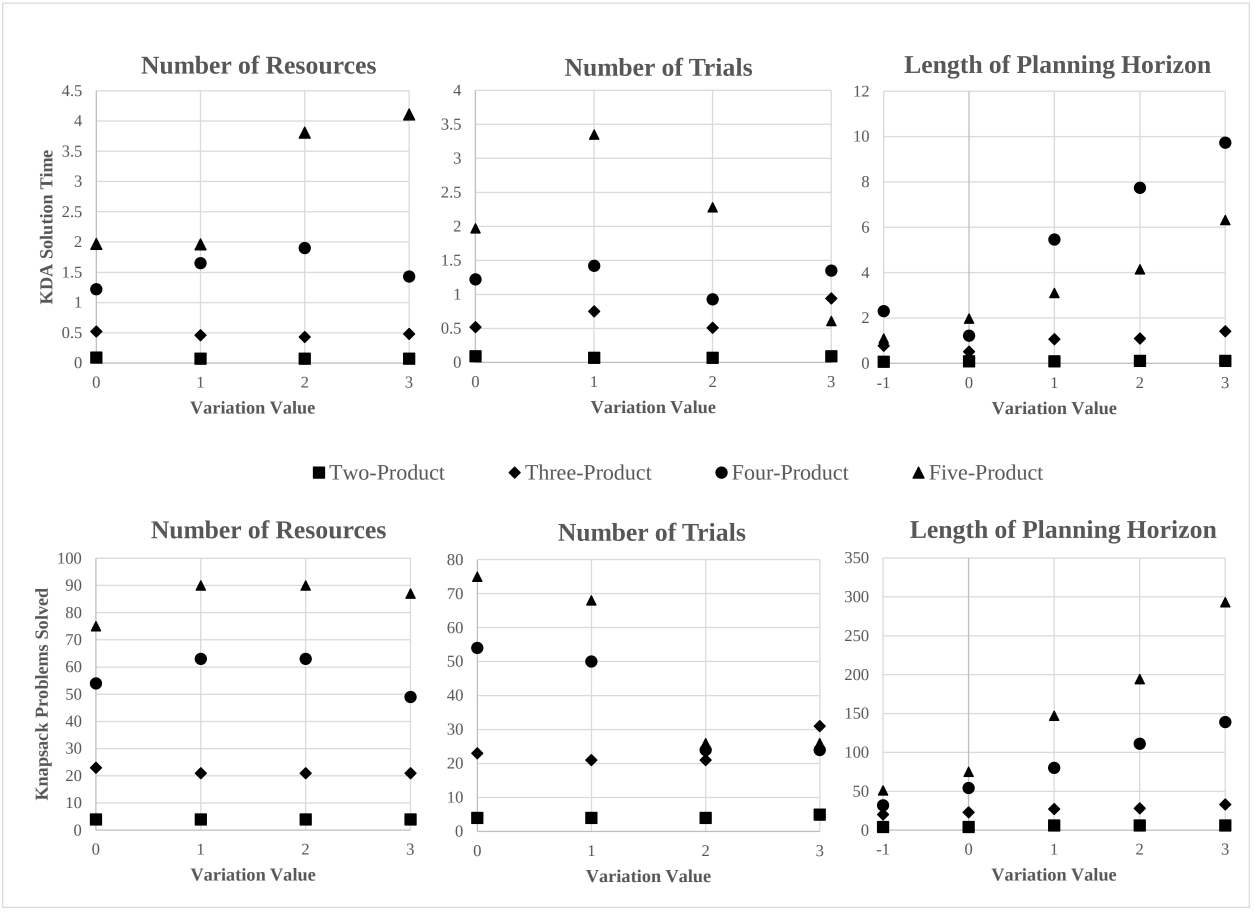


Figure 5.7 The impact of problem size on the KDA solution time and the number of knapsack problems

The knapsack problem is an np-complete problem, and, hence, the solution time for each knapsack problem increases in non-polynomial time based on the number of items and the dimensionality of the problem (number of weight constraints). The KDA solution time is equal to the cumulative solution time of all the knapsack problems plus the additional time consumed for logic operations (e.g., for determining eligible items). Figure 5.8 plots the KDA solution time against the number of knapsack problems solved. The plot suggests that the KDA time complexity is linearithmic ($\mathbf{O}(n\log(n))$) where n is defined as the number of knapsack problems solved. In Christian and Cremaschi (2015), we presented a loose theoretical upper-bound on the number of knapsack problems that may be solved. For all problems considered in this computational study, the actual number of knapsack problems solved are significantly lower than the theoretical bound. Improving the theoretical bound on the number of knapsack problems solved will allow for better prediction of KDA solution times.

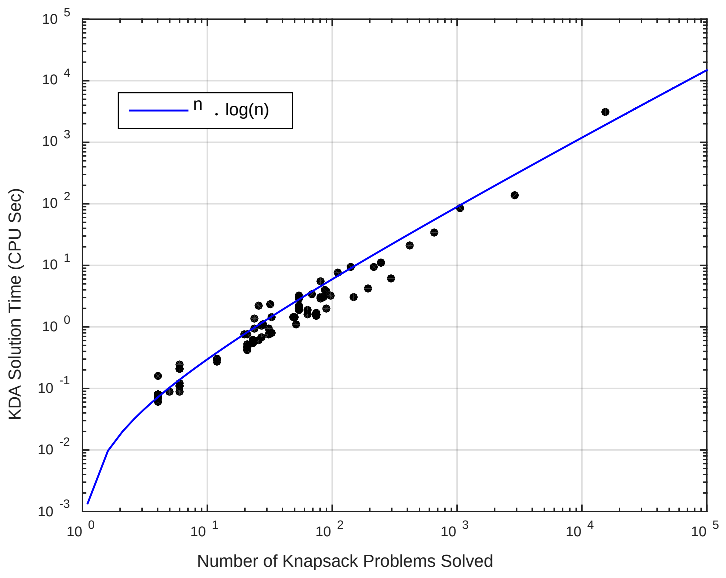


Figure 5.8 The correlation between number of knapsack problems solved and the solution time in CPU seconds of the KDA

The number of knapsack problems generated depends on the number of realizations that occur in the planning horizon and the number of items. The number of realizations in the planning horizon is affected by the length of the planning horizon, and the duration of each clinical trial that is selected. A longer planning horizon translates to more realizations and thus more knapsack problems. Quantifying the number of realizations also requires the knowledge of packed items which is not known *a priori*. Each item packed in the knapsack has a corresponding trial duration. The duration for each trial is not guaranteed to be identical, therefore more realizations will occur when items with short trial durations are packed. The packed items in knapsack problems are limited due to resource constraints. However, resource constraints only constrain the number of resources available not the number of items packed thus some knapsacks may have many light (low resource) items while others just a few heavy (high resource) items. The bound presented in Christian and Cremaschi (2015) takes a very conservative approach to addressing each of these complications. The bounding approach assumes each knapsack packs a maximum number of items based on the items weights (resource requirements) and the maximum number of resources available. It also assumes that the same knapsack is packed at each realization regardless of the availability of items to pack and outcomes of items. Realizations are assumed to occur every $\mathbf{min}(\tau_{d,j}\ \forall d, j)$ time periods. These assumptions cause the bounding approach to calculate the number of knapsacks solved if the maximum number of items allowed by the available resources is packed in every knapsack at each realization.

Here, we introduce an algorithm that improves the bound by considering that each branch in the decision tree is limited to packing each item once. It assumes that realizations occur when the first item packed in the knapsack is completed. Similar to the previous approach, the algorithm does not consider the value of the uncertain parameter after it has been realized. It assumes that subsequent items can be packed whether or not the clinical trial is successful. This assumption creates an upper bound on the number of knapsack problems solved. The algorithm is presented in Eq 5.18.

```
$A = \emptyset$
$tp = 0$
$EKS = 1$
$while\ tp < |\boldsymbol{T}|\ do$
        $C = \boldsymbol{I} \backslash A$
        $B = \text{solution}(\max(\sum x_i \ \ s.t.\ \sum w_i x_i \ \le \rho^{max}\ \forall i \in C\ ))$        (5.18)
        $A = A \cup B$
        $tp = tp + \min(\tau_{d(i),j(i)}\ \forall i \in B)$
        $EKS = EKS + EKS \cdot 2^{|B|}$
$end$
```

The algorithm calculates the estimated number of knapsacks (EKS) by stepping through the planning horizon. At each step, the algorithm finds the maximum number of items that can be packed assuming that items that have been packed previously are no longer eligible. The algorithm is applied to the original set of problems presented in Christian and Cremaschi (2015). For the two-product case, it predicted a total of five knapsack problems. Using the KDA approach, the number of knapsack problems solved was four. The difference is due to the generation of knapsack problems regardless of the trial outcomes. In the two-product case, the number of failures that result in counting fictitious knapsack problems is low. This count increases with the size of the problem (number of products and trials). Therefore, as the size of the problem increases the estimate provided by Eq. 5.18 degrades. For the three product case, the ratio of the number of actual knapsack problems solved to the estimated number is 0.20. For the largest problem (the ten-product case), the ratio is the lowest at 0.05.

Another measure of problem size is the number of scenarios, and Figure 5.9 shows the change in the number of knapsack problems solved with the number of scenarios. Notice that the general trend of Fig. 5.9 shows an increase in the number of knapsack problems solved as the number of scenarios increases. The number of scenarios does not directly impact the number of knapsack problems solved, however the number of scenarios in a problem does depend on the number of uncertain parameters and the number of realizations for each uncertain parameter. The number of uncertain parameters in pharmaceutical R&D pipeline management problem depends on the number of products. For each additional product, there is one additional uncertain parameter, and for each additional clinical trial, there is one additional realization per uncertain parameter. For each additional uncertain parameter, there are $(|\boldsymbol{J}| + 1)$ times more scenarios. Similarly, for each realization, there are $\left(\frac{|\boldsymbol{J}|+1}{|\boldsymbol{J}|}\right)^{|\boldsymbol{D}|}$ times more scenarios. Both the number of knapsack problems and the number of scenarios increase with increases in number of products, and hence, the positive trend in Fig. 5.9. Unlike increasing the products, increasing the number of clinical trials does not have a significant effect on the number of knapsack problems solved. In Fig. 5.9, this phenomenon is observed with the points that sit on an almost horizontal line at 28 knapsack problems. This horizontal line represents an increase in scenarios without an increase in the number of knapsack problems solved. At 1024 scenarios, Fig. 5.9 shows that the number of knapsack problems solved increases from 24 to 559 without an increase in the number of scenarios. This occurs when the length of the planning horizon is increased. Increasing the length of the planning horizon neither adds new uncertain parameters nor adds new realizations, thus not impacting the number of scenarios. However, it increases the number knapsack problems solved because there are more decision points.

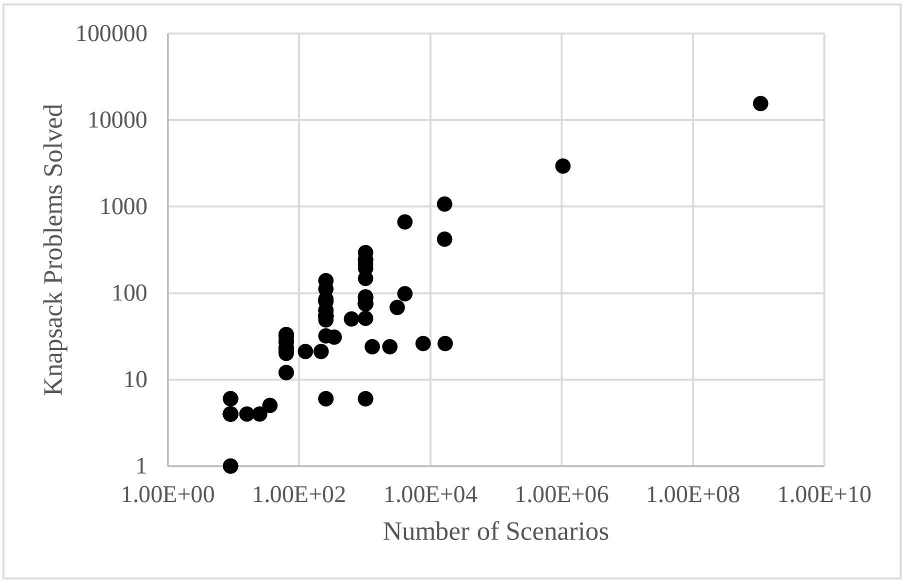


Figure 5.9 The number of knapsack problems solved using the KDA plotted against the number of scenarios in the equivalent MSSP

5.6.3 Impact of the Proposed Knapsack Problem Generation Rules

The ENPVs and solution times for the KDA using all three knapsack problem generation rules are summarized in Table 5.3. For completeness, the same information is provided for the solution of the rigorous MSSP. The solution quality does not change significantly among different knapsack problem generation schemes (Table 5.3). All ENPVs obtained using the KDA approach remain within 5% of the rigorous MSSP ENPVs.

Table 5.3 Solution times and percent error for original knapsack, AER, and ETP knapsack generation schemes

	MSSP		Original KDA		Every Time Period (ETP)		After Every Realization (AER)	
	ENPV	*Solve Time (CPU sec)*	*Percent Error*	*Solve Time (CPU sec)*	*Percent Error*	*Solve Time (CPU sec)*	*Percent Error*	*Solve Time (CPU sec)*
two-product	1110	0.02	1.17	0.05	1.17	0.09	1.17	0.07
three-product	1189	0.84	0.93	0.37	5.05	0.96	1.51	0.35
four-product	1683	2.71	0.42	0.81	0.30	1.26	0.30	0.87
five-product	2083	16.95	1.92	1.24	1.44	2.12	1.44	1.7
six-product	2412	97.67	0.37	1.63	0.21	3.18	0.21	2.45
seven-product	--	--	--	8.17	--	21.33	--	16.67
Average			***0.96***		***1.63***		***0.93***	

The ENPVs obtained by the ETP and the AER knapsack generation rules are identical for all but the three-product case. The difference in the three-product case is due to the lack of realizations at every time period. The decision trees obtained by the KDA using each of the knapsack problem generation schemes and from the solution of the rigorous MSSP for this case are given in Fig. 5.10. In the figure, the nodes correspond to decisions. The information in the parentheses represent the (drug, trial) selected at the decision point.

Visually the decision trees obtained using the original KDA (Fig. 5.10(a)) and the AER decision rule appear (Fig. 5.10(b)) sparser than the MSSP decision tree (Fig. 5.10(d)). In contrast, the decision tree for the ETP approach (Fig. 5.10(c)) is denser than the MSSP decision tree. The ETP approach results in the densest decision tree because it generates new knapsack problems at each time period. The solutions of these knapsack problems, in turn, may result in investment decisions at each time period. For the three-product case, the ETP approach overschedules the trials compared to the original KDA, partly due to how the resource overscheduling constraint functions. This constraint ensures that the total number of resources needed to complete the remaining clinical trials without delay is less than the available number of resources during that same time period. This is implemented using a cumulative approach. For instance, consider a product that requires one resource for the first trial, two resources for the second trial, and three resources for the third trial. The total number of resources needed to complete all three trials is six. The cumulative approach ensures that for the duration of product's clinical trials there are six resources available. It does not account for the fact that the first trial only requires one resource while the third trial requires three. Because the resource requirements for each trial are not differentiated, the algorithm may run into instances where the cumulative number of resources needed is available but the resources needed to start a particular trial at a specific time are not available. Solving problems at every time period provides more opportunities for this behavior.

Figure 5.10(b) shows the decision tree generated using the AER approach has more decision points than the original KDA decision tree but fewer than the MSSP decision tree. Notice that despite considering new knapsack problems at each realization, the approach fails to obtain the first two decisions correctly, whereas the ETP approach captures this behavior. The AER approach is unable to do so because knapsack problems are only generated after realizations.

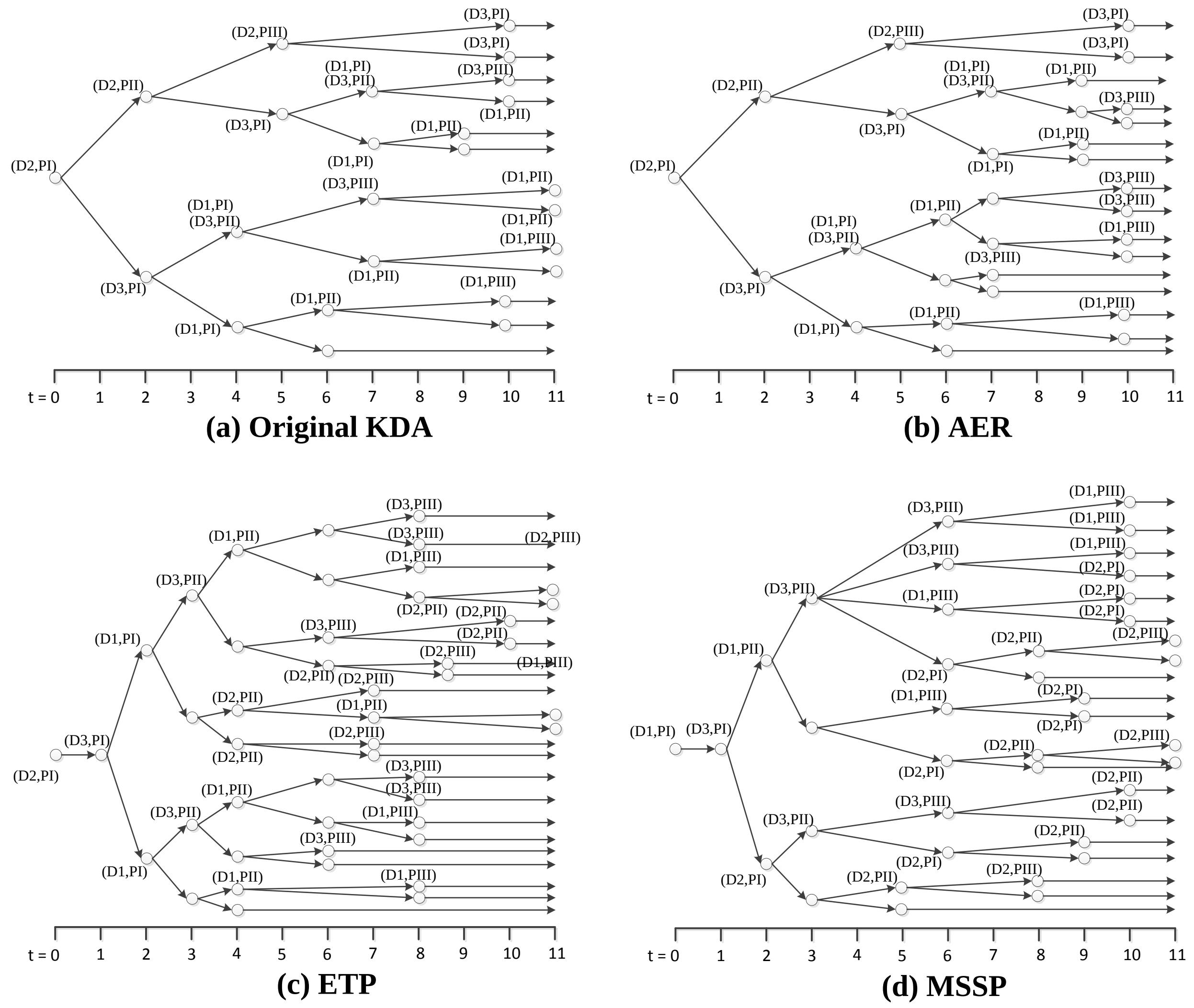


Figure 5.10 The decisions trees obtained by KDA for the three product case using (a) the original KDA (Christian and Cremaschi, 2015), (b) the AER, and (c) ETP knapsack generation schemes; and (d) the MSSP (Christian and Cremaschi, 2015) decision tree.

The KDA solution times are orders of magnitudes faster than the time required to solve the deterministic equivalent MSSP for all variations of the knapsack sub-problems generation schemes (Table 5.3). When solution times among different knapsack problem generation schemes are considered, in general, the original KDA yields the solutions the fastest and the KDA using the ETP approach the slowest. The only exception is the solution times for the three-product case in which the times are small enough that the measurement error may exceed the measured time. In the original KDA, fewer knapsack problems were generated because all realizations are required to occur before new problems are generated. With both proposed methods, more knapsack problems are generated. Figure 5.11 shows the total number of knapsacks solved using each of the knapsack generation approaches. The problems used are the original problems from Christian and Cremaschi (2015). The computational studies on problem size revealed that the solution time increases with the number of knapsack problems generated. Notice that for the two-product and three-product cases, the differences between the number of knapsack problems solved are small. In the two-product case, the difference between the number of knapsack problems generated in the original KDA (least) and the ETP approach (most) is only two. In the larger cases (six- and seven-products), the difference in the number of knapsack problems solved increases considerably. In the seven-product case, the difference between the number of knapsack problems generated in the original KDA algorithm (least) and the ETP approach (most) was 462. For all cases, the AER approach generated more knapsack problems than the original KDA but as many or fewer than ETP scheme. In the cases where the realizations for decisions occur at each time period, generating knapsack problems using the ETP scheme is identical to generating knapsack problems using the AER scheme.

5.6.4 Impact of Different Formulations of Resource Overscheduling Constraint

The resource overscheduling constraint implemented in the original KDA tries to ensure that there will be resources available to start the subsequent clinical trial if a clinical trial is successful. Using the six problems originally solved in Christian and Cremaschi (2015), the impact of different formulations for avoiding resource overscheduling are investigated using: (1) the original resource overscheduling constraint, (2) a penalty term in the objective function, and (3) a probabilistic resource overscheduling constraint.

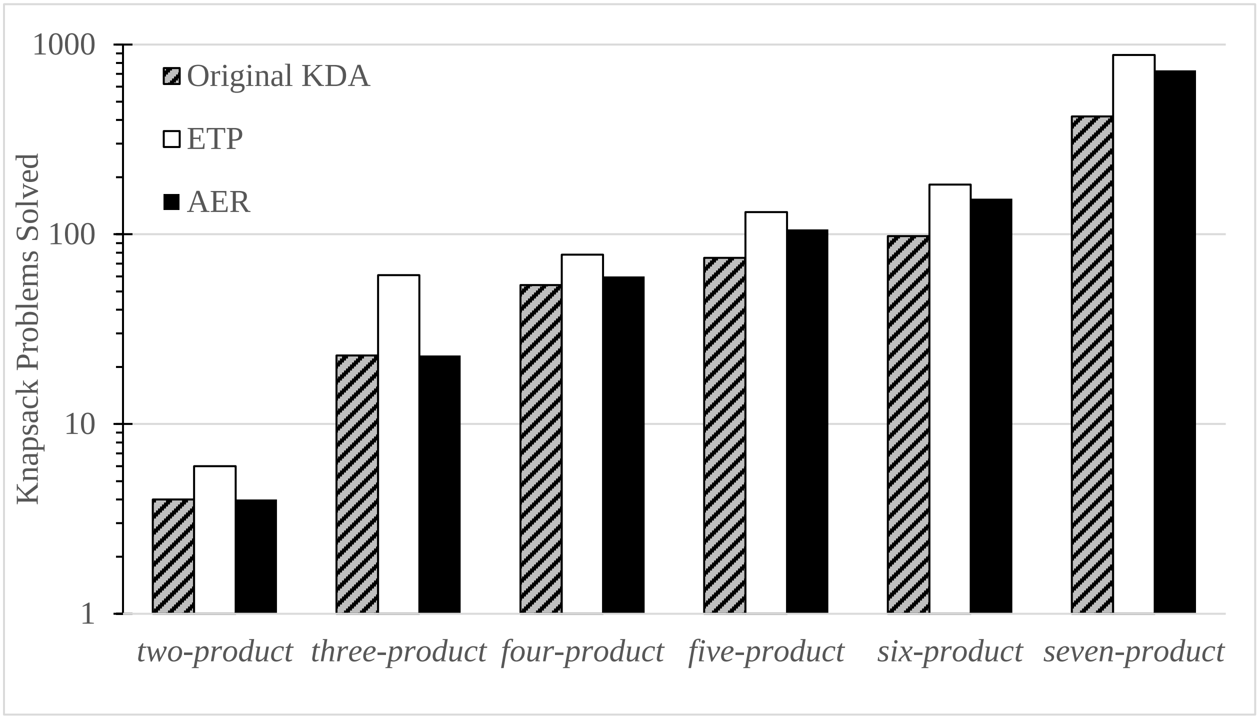


Figure 5.11 The number of knapsack problems solved using the original KDA, the ETP, and the AER knapsack generation approaches.

We varied the penalty coefficient between 1 and 500 and solved each of the base case problems. Varied values of the penalty coefficient resulted in solutions with identical equivalent ENPVs. A closer investigation of the decision trees generated using different values of the penalty coefficient revealed that they did not change for the range considered. This suggests that investments in additional products is primarily limited by the availability of 'here-and-now' resources.

The results for the implementation of the probabilistic approach are summarized in Table 5.4. Comparing the objective function values in Table 5.4 reveals that values are lower when the probabilistic overscheduling constraint is used in all but the seven product case. In the seven product case, the improved value suggests that the value of allowing additional items to be packed outweighs the losses incurred for products being idle in the pipeline.

Table 5.4 The objective functions and solution times of the probabilistic resource constraint and the original resource constraint

	Original KDA Sub-Problem Generation			
	Probabilistic Resource Overscheduling Constraint		**Original Resource Overscheduling Constraint**	
	Objective Value	Solve Time (CPU Seconds)	Objective Value	Solve Time (CPU Seconds)
two-product	1097	0.05	1097	0.05
three-product	1165	0.47	1178	0.37
four-product	1665	1.17	1676	0.81
five-product	2038	2.06	2043	1.24
six-product	2399	2.86	2403	1.63
seven-product	2871	17.48	2870	8.17

The KDA solution times for the proposed approaches to formulating resource overscheduling constraint are similar to that of the original one for the smaller problems (two- and three-product cases). For the larger problems (six- and seven-product cases), there is a noticeable increase in the solution times. For the penalty approach, this increase stems from the increase in the solution times of individual knapsack problems. The time for solving knapsack problems with the penalty term in the objective function was twice as long as solving the original knapsack problems. In each of the case studies, the average time needed to solve a knapsack problem with the penalty term was 0.04 CPU seconds whereas solving the original knapsack problems required 0.02 CPU seconds. In the smaller cases, the number of knapsack problems solved is small, and thus, the difference in the total solution times is insignificant. In larger cases, however, the total number is large, which leads to longer overall solution times for the penalty approach. The difference in the time to solve each knapsack problem with the penalty term can be attributed to the additional variables that were added to linearize the penalty term in the objective function.

Similar to the penalty approach, the probabilistic approach also shows increased solution time for larger problems (six- and seven-products). The knapsack problems with the probabilistic resource overscheduling constraint use a similar formulation to the original knapsack problems. For smaller problems (two- and three-products), the solution times for the knapsack problems with the probabilistic constraint and the original knapsack problems are identical (0.02 CPU seconds). The larger problems revealed that the knapsack problems with probabilistic constraint take longer to solve (0.03 CPU seconds). The linear relaxation of the knapsack problem with the probabilistic constraint yields a larger feasible region than the original knapsack problem, which yields a looser upper bound. Therefore, for larger instances of the problem with more items that can be packed in a knapsack (i.e., products), it takes longer to solve the problems with the probabilistic constraint.

5.7 Generalization of the Knapsack Decomposition Algorithm

The knapsack decomposition approach efficiently provided tight feasible bounds for all studied instances of the clinical trial planning problem. We hypothesize that the KDA would provide tight bounds for other MSSPs. Here, we extend the KDA to solve the NTIP problem. A complete description of the MSSP formulation for the NTIP problem can be seen in Section 3.2.3.

The clinical trial planning problem consists of an objective function dependent on a set of discrete decisions. The decisions are constrained by a set of resources. In the construction of the KDA for the clinical trial planning problem, the discrete decision variable (whether each clinical trial is started at each time period) is converted into a set of items. The value of the items roughly corresponds to the expected value of the decision. The weights correspond to the resource requirements of the decision variables.

The first step of the KDA algorithm is the identification of items of the knapsack problems. For the clinical trial planning problem, the items corresponded to the discrete decision variables. The NTIP problem has continuous decision variables. If the items in a knapsack problem are continuous, the knapsack is a special case of the knapsack problem, where the true solution can be found using a greedy algorithm. Recalling from the formulation, the NTIP problem has three decision variables: the level of production in each technology ($M_{i,n,t,s}$), the level of capacity expansion of each technology ($X_{i,t,s}$), and the level of research expenditures ($RD_{i,t,s} - RD_{i,t-1,s}$). In addition to the decision variables, the NTIP problem also has non-trivial recourse actions. After the realization of uncertainty (i.e. yield) there may be a need to purchase additional product to assure that demand is met. In contrast, the clinical trial planning problem has trivial recourse actions.

To address the differences in recourse actions, modifications to the KDA are required. Figure 5.12 shows an overview of the modified KDA. From this point forward, we refer to the generalized version of the KDA where the decision variables can be either continuous or discrete and the recourse actions are non-trivial as the Expected Value Decomposition Algorithm (EVDA). The algorithm begins by setting the time period t equal to zero and storing information on the initial sub-problem. At $t = 0$, there is a single sub-problem. In addition to the set of sub-problems at time t, $k_{n,t} \in N(t)$, there exists a set R, which represents the realizations that have occurred prior to time period t for sub-problem $k_{n,t}$. At t = 0, R is the empty set. This is consistent with the case where no uncertainty has been realized. The algorithm then generates and solves each of the sub-problems $k_{n,t} \in N(t)$ at time t. After solving each sub-problem, the EVDA realizes the uncertainty associated with the decisions made in the sub-problem, E_n. From the realized uncertainty E_n, a set Θ is constructed. The set Θ represents the unique realizations of uncertainty for the realized parameters E_n. For each unique realization of uncertainty $m \in \Theta$, sub-problems at time $t + 1$ are constructed $N(t + 1)$. The recourse actions for each unique realization of uncertainty are calculated. The recourse actions represent actions taken after the realization of uncertainty in time t but prior to decisions made at time period $t + 1$. After solving all sub-problems at time t, the algorithm increments the time and repeats the process of generating EVDA problems, solving EVDA problems, realizing uncertainty, and determining recourse actions. The process continues until the end of the planning horizon is reached.

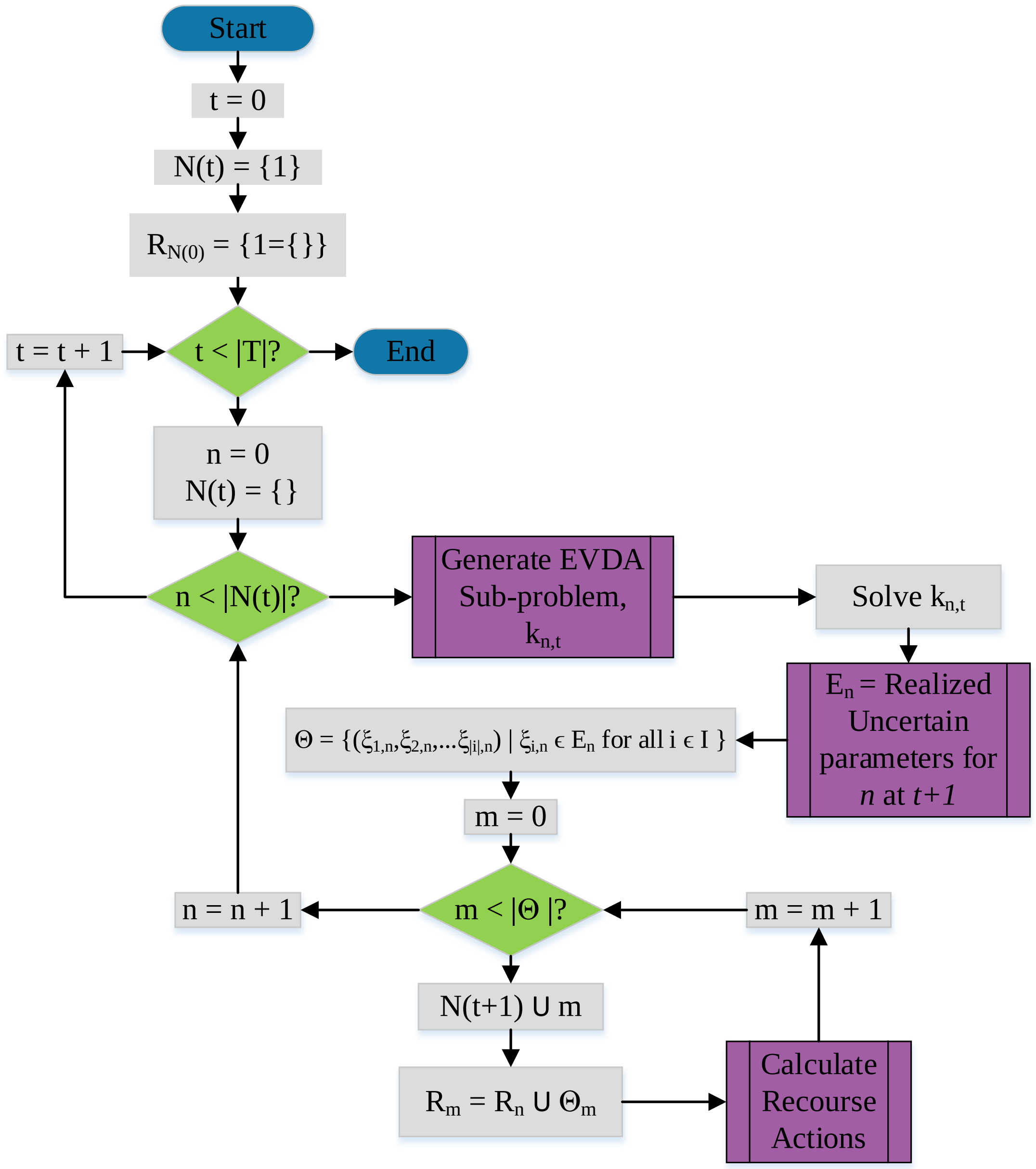

Start
$t = 0$
$N(t) = \{1\}$
$R_{N(0)} = \{1=\{\}\}$
$t = t + 1$
$t < |T|?$
End
$n = 0$
$N(t) = \{\}$
$n < |N(t)|?$
Generate EVDA Sub-problem, $k_{n,t}$
Solve $k_{n,t}$
E_n = Realized Uncertain parameters for *n* at *t+1*
$\Theta = \{(\xi_{1,n},\xi_{2,n},\ldots\xi_{|i|,n}) \mid \xi_{i,n} \in E_n \text{ for all } i \in I \}$
$m = 0$
$n = n + 1$
$m < |\Theta|?$
$m = m + 1$
$N(t+1) \cup m$
$R_m = R_n \cup \Theta_m$
Calculate Recourse Actions

Figure 5.12 The EVDA algorithm.

5.7.1 Formulation of the Knapsack Sub-Problem for the NTIP Problem

The EVDA sub-problem is constructed as a knapsack problem. If the decision variables are continuous, the knapsack problem is a continuous knapsack problem. For the NTIP problem, the decision variables $X_{i,t,s}$, $M_{i,n,t,s}$, and $(RD_{i,t,s} - RD_{i,t-1,s})$ will be represented as the items in the continuous knapsack problems . The next step is to determine the value of each item. In general, the value of the item is the expected contribution to the objective function for making the decisions associated with the item at a particular time period t. For the NTIP problem, the objective function minimizes the cost of production where the cost of production at each time period is defined using Eq. 5.19 where CC_i is the cost of capacity expansion, $MCst_n$ is the purchase cost of chemical n, RDX_i is the level of research spending, and F_n is the amount purchased of chemical n.

$$\sum_n MCst_n \cdot F_n + \sum_i CC_i X_i + \sum_i RDX_i \tag{5.19}$$

Notice that the variable F_n is not a decision variable. Instead, F_n is a derived from the decision variable $M_{i,n,t,s}$ in the MSSP. To better visualize the derivation of the EVDA sub-problem we include Eq. 5.20.

$$F_n = D_n - \sum_i \gamma_{i,n,PR} \chi_i M_{i,PR} \tag{5.20}$$

Equation 5.20 represents the mass balance where the amount purchased at each time period depends on the level of demand and on the amount produced. Substituting Eq. 5.20 into Eq. 5.19 results in Eq. 5.21.

$$\sum_n MCst_n \cdot \left(D_n - \sum_i \gamma_{i,n,PR} \chi_i M_{i,PR} \right) + \sum_i CC_i X_i + \sum_i RDX_i \tag{5.21}$$

Equation 5.21 naturally represents the contribution of each decision, and the corresponding item, to the objective function value, therefore its expected value is used as the objective function of the knapsack sub-problems. The objective function for the NTIP knapsack sub-problems is given in Eq. 5.22.

$$\min E \left[\sum_n MCst_n \cdot \left(D_n - \sum_i \gamma_{i,n,PR} \chi_i M_{i,PR} \right) + \sum_i CC_i X_i + \sum_i RDX_i \right] \tag{5.22}$$

Since expected value is a linear operator, equation can be reduced to,

$$\min \sum_n MCst_n \cdot \left(E[D_n] - \sum_i \gamma_{i,n,PR} \cdot E[\chi_i] \cdot M_{i,PR} \right) + \sum_i E[CC_i] \cdot X_i + \sum_i RDX_i \quad (5.23)$$

The values of expected demand ($E[D_n]$), and expected yield $E[\chi_i]$, and $E[CC_i]$ are straight forward to calculate. As with any expected value calculation, the equivalent value is equal to the probability of the outcome occurring multiplied by the value of the outcome. In the case of $E[CC_i]$, the value of the outcome depends on the variable X_i. As such, the equation is incorporated as a constraint to the EVDA sub-problems. This constraint is shown in Eq. 5.24. In Eq. 5.24, $CX_{i,t-1}$ and $RD_{i,t-1}$ represent the cumulative installed capacity and research investment at the previous time period. In the case of the EVDA solution, this value may be calculated as the sum of X_i and RDX_i for all parent EVDA sub-problems which have been solved previously.

$$E[CC_i] = \sum_{a \in \alpha} \sum_{b \in \beta} P\big((a,b)\big) \cdot CC0_i \cdot \left(\frac{CX_{i,t-1} + X_i}{CX_{i,0}} \right)^b \cdot \left(\frac{RD_{i,t-1} + RDX_i}{RD_{i,0}} \right)^a \quad (5.24)$$

Any constraint scenario specific constraint of the MSSP persists as a constraint in the EVDA sub-problem. In the NTIP problem, these are shown in Eq 5.25–5.29. Notice that when uncertain parameters appear in a constraint they are treated identical to that of the objective function (i.e., the expected value is substituted)

$$M_{i,n} \le Y_{i,3}\big(CX_{i,t-1} + X_i\big)\, E[\theta_i]\ \forall i, n \quad (5.25)$$

$$CX_{i,t-1} + X_i \ \ge CX_{i,sg}{}^{min} \cdot Y_{i,sg} \qquad \forall i, sg \quad (5.26)$$

$$CX_{i,t-1} + X_i \ \le \sum_{sg\,<\,3} \big(CX_{i,sg+1}{}^{min} - CX_{i,sg}{}^{min}\big) Y_{i,sg} \qquad \forall i \quad (5.27)$$

$$Y_{i,sg} \le Y_{i,sg-1} \qquad \forall i, sg > 1 \quad (5.28)$$

$$Y_{i,sg} \leq Y_{i,sg,t-1} \qquad \forall i, sg \tag{5.29}$$

Similar to the $CX_{i,t-1}$, the value of $Y_{i,sg,t-1}$ is calculated using the solutions from the previous EVDA sub-problem. The overall formulation for the EVDA sub-problem for the NTIP problem includes Eqs. 5.23-5.29. In addition, the problem bounds the value of X_i between 0 and X_i^{max}, and RDX_i between 0 and RDX_i^{max}. The resulting sub-problem is a mixed integer non-linear program (MINLP).

5.7.2 Case Studies

The EVDA was implemented using Python 3.5.2 and Pyomo 5.1, and it uses BARON 17.1 (Tawarmalani and Sahinidis, 2005) to solve MINLP sub-problems to 0.1% optimality. We used the EVDA to generate bounds on a total of four NTIP case studies. The case studies are identical to the ones described in Section 3.2.4. The first case study considers the comparison of a mature technology and an undeveloped technology. Investment decisions are made on a planning horizon of three years divided into one-year time periods.

For Case Study 1, the EVDA solved 217 MINLP sub-problems to optimality generating a feasible solution with an objective value of 151.8 billion dollars. Each sub-problem required an average of 0.01 CPU seconds to solve and resulted in a total algorithm time of 40 CPU seconds. The linear relaxation of the case study has an objective function value of 146.4 billion. The bounds provided are tighter than the bounds obtained by solving the NLP after fixing the integer variables to the value of the MILP solution (188.2 billion). The relative gap between the EVDA solution and the linear relaxation model of the MSSP is 3.5%. In comparison, the relative gap between the linear relaxation solution and the NLP solution fixing the integers from the linear relaxation solution is 22.1 %. The experimental cumulative distribution function (ECDF) for the linear relaxation of the MSSP, the EVDA, and NLP are shown in Fig. 5.13.

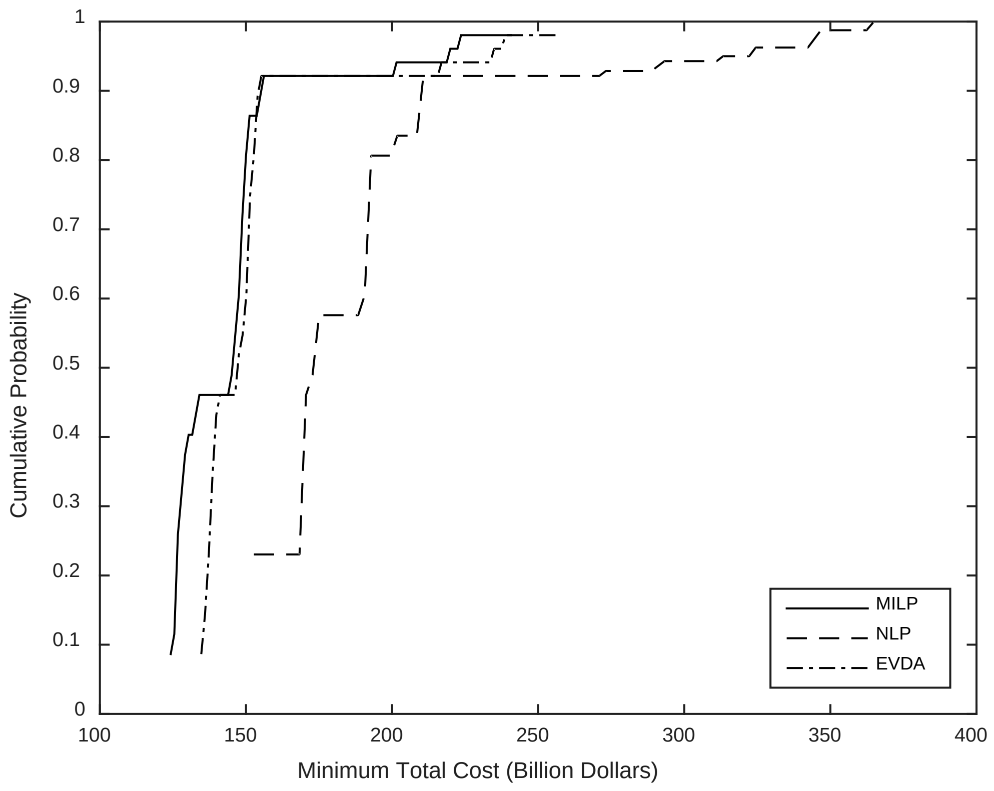


Figure 5.13 The Experimental CDF (ECDF) for the Linear Relaxation (MILP) of the MSSP, the NLP fixing the integers from the linear relaxation solution, and the EVDA algorithm.

The ECDF shown in Fig. 5.13 shows that the MILP solution for the MSSP a lower total cost for every scenario compared to the NLP and the EVDA. This type of solution is expected as the non-linearity in the cost of capacity expansion is under-estimated by the upper and lower estimators. The EVDA algorithm provides a solution which is superior to the solution obtained by solving the NLP obtained by fixing the integer variables to the value of the MILP solution.

At time period one, the primary difference between the MILP solution and the feasible solution provided by the EVDA is the investment in research expansion. The value for the research expansion at time period one of the MILP solution is 1 million. In contrast, the EVDA algorithm suggests a research investment of 10 million dollars (the maximum allowable). Since the EVDA solves the MINLP, the cost of capacity expansion is not estimated with linear lower and upper estimators. An investment in research indicates that the value of the investment (10 million dollars) is worth the reduction in capacity cost. Since the cost is approximated in the linear relaxation of the MSSP, the value of research is underestimated.

Capacity expansion at the first time period also differs. The EVDA recommends installing 6 Mtonnes per technology despite the possibility of TECH2 failing to be developed. The expected success (i.e. the joint probability of successfully completing all stages) of developing TECH2 is greater than 95%. The EVDA determines investment decisions based on the expected outcomes. In this case study, the expected success is high enough to justify a full development of TECH2. The MILP solution suggests a more conservative approach. It recommends installing 6 Mtonnes of capacity for TECH1 (the developed technology) and 2 Mtonnes for TECH2.

At time period two, the solution provided by the EVDA recommend maximum research and capacity expansion investment in TECH2 if the development of TECH2 is successful. If the development of TECH2 is unsuccessful, the algorithm suggests installing 6 Mtonnes of TECH1 capacity. The MILP solution is more conservative. The level of capacity expansion of TECH2 depends not only on whether or not it is successful but also on the realized values of the uncertain parameters (α the R&D elasticity, β the capacity expansion elasticity, and χ the yield of the process). Assuming that the development of TECH2 is successful, in the scenarios where the elasticities and yield are the highest (corresponding to the lowest cost of capacity expansion), the investment in capacity expansion for TECH2 is highest at 5.885 Mtonnes. In contrast, the scenarios with the lowest elasticities and yield only install 5.644 Mtonnes of TECH2 capacity. Expansion of TECH1 is identical to time period 1 in the linear relaxation solution. The solution recommends installing 6 Mtonnes (the max allowable expansion).

At time period three, in the scenarios where the development of TECH2 is successful both the MILP solution and the EVDA solution can expand the capacities of the technologies to meet the entire demand. In these scenarios, the difference between the expected demand and amount that can be produced with the capacity levels from time period two, is such that the installation of additional capacity from a single technology is sufficient. Both algorithms select to install capacity for TECH2. The level of installation depends on the realized yield value but is approximately 2 Mtonnes. In the scenarios (both the linear relaxation of the MSSP and the EVDA) where TECH2 is not successful, 6 Mtonnes of TECH1 capacity is installed.

We solved the competing technologies case study (Case Study 2), the retrofitting case study (Case Study 3), and the biomass to commodity chemical (BTCC) case study using the EVDA. The case study specific parameters can be found in Appendix B. A complete discussion of the results for the MSSP problem can be found in Section 2.3.4. In each case study, we were unable to find a feasible solution due to the computational complexity of the NTIP problem formulation.

The EVDA yielded a feasible solution in 1.1 CPU seconds for case study 2 which compares two undeveloped technologies TECH1 and TECH2. The algorithm solved four MINLP sub-problems to an optimality gap of 0.1%. Each sub-problem required 0.12 CPU seconds to solve. The solution for Case Study 2 bounded the problem with a 9% gap. Overall, the decisions in the MILP solution and the EVDA solution differed substantially. The EVDA recommended no investment in either technology. In contrast, the linear relaxation solution had invested in both technologies. The investment decisions between the MILP soltuion and the EVDA differ due to the approximation of the capacity expansion cost. In the linear relaxation solution, the capacity expansion cost is under-estimated thus the value of expanding the undeveloped technologies is higher. The probability of success is not high enough to offset the potential loss in the EVDA resulting in a do-nothing solution.

Case Study 3 considers retrofitting an existing production network. Two existing technologies can produce CHEM4 (Figure 3.10(B)). TECH1 converts CHEM1 to CHEM3 and TECH3 converts CHEM3 to CHEM4. The problem has two potential new technologies. If successfully developed, TECH2 can be used to convert CHEM2 to CHEM3. Alternatively, there exists the possibility that if successfully developed

TECH4 can be used to directly convert CHEM1 to CHEM4. The parameters for the NTIP MSSP are given in Appendix B. The case study had an EVDA objective function value of $6.884 billion which bounds the problem with an 0% percent gap. The EVDA and MILP solution to Case Study 3 are identical. In both solutions, the optimal investment strategy is to use existing capacity to produce the chemical. The EVDA algorithm was able to find the optimal solution in 0.60 CPU seconds by solving eight MINLP sub-problems. Sub-problem solution times averaged 0.02 CPU seconds. The difference in the solution times between Case Study 1, Case Study 2 and Case Study 3 demonstrate that the solution time of the EVDA algorithm depends on the number of sub-problems. The number of sub-problems solved during the implementation of the EVDA algorithm is a function of the uncertainty realizations that occurs during the planning horizon.

The final case study considered was the BTCC case study, which consists of a process flow network where biomass can be fermented to ethanol and can be dehydrated to ethylene. Alternatively, ethylene can be produced by cracking naphtha. The problem considers a new production route consisting of two undeveloped technologies. The first is a gasification process which can potentially convert biomass to syngas. The second is a catalytic conversion technology which may be used to convert syngas to ethylene if successfully developed. When applied to the BTCC case study, the EVDA obtained a solution of $168 billion dollars. The linear relaxation of the MSSP provided a solution of $161 billion. This results in an optimality gap of 4.1%. To obtain a solution, the EVDA solves 19 MINLP sub-problems. Each sub-problem required 0.03 CPU seconds to solve. The total runtime of the EVDA was 1.68 CPU seconds. The EVDA solution is very similar to the MILP solution. In the MILP solution, ethylene is only produced using the cracking technology. Any ethylene that cannot be produced by cracking naphtha is purchased. No capacity expansion in any technology occurs. The EVDA solution also recommends not installing any new capacity. However, the EVDA does recommend using existing fermentation and dehydration technologies to produce ethylene from biomass.

5.7.3 Conclusions

In this work, we extended the knapsack-problem based decomposition algorithm to solve MSSPs with continuous variables and non-trivial recourse actions. The EVDA provided tight feasible solutions for the each of the NTIP case study problems considered. In Case Study 1, the EVDA provided a solution which was superior to the NLP solution where the integer variables were fixed to their MILP solution values. Case study 2 was bounded at 9% percent using the EVDA solution. In Case Study 3 where there is no capacity expansion, the EVDA confirmed that the MILP solution was the optimal one. The runtime of the EVDA algorithm depends on the size of the decision tree. In cases where there are few realizations of uncertainty (i.e., a small decision tree) the EVDA yields a feasible solution very quickly. However, when the decision tree is larger, the solution time grows as a function of the number of sub-problems solved. Depending on the size of the MINLP solved, this time may become prohibitive. Future work may investigate using approximation approaches to solve the MINLP sub-problem.

CHAPTER 6
A Branch and Bound Algorithm to Solve Large-Scale Multi-Stage Stochastic Programs with Endogenous Uncertainty

6.1 The Branch and Bound Algorithm

The general algorithm is summarized in Fig. 6.1. At the initialization step, the values for the relative gap between the dual bound and the primal bound (α), the tolerance (ε), and the maximum number of iterations (i^{max}) are set. Next, the iteration count, i, is set to zero. The algorithm starts by generating a feasible solution, φ_i, using the KDA, which is a heuristic algorithm that solves the original MSSP by decomposing it into a series of knapsack problems. The Equivalent Expected Net Present Value (EENPV) for the KDA solution provides the initial primal bound, PB_0. The algorithm next determines the branching variable(s) by comparing the values of decision variables that have been fixed in the current branch to values of decision variables in the KDA solution, φ_i. From the decision variables in the KDA solution (φ_i) that have not yet been fixed in the current branch, the ones that occur at the earliest time period are selected as the branching variable(s).

Assuming that there is one binary branching variable, two new optimization problems are generated; in one, the branching variable takes the value of one, in the other, the value of zero. In both problems, the values of decision variables that were fixed in the parent branch are carried over. The solution of each of these optimization problems provides a dual bound, D_n, for each branch n. The optimization problems are added to the set of active branches, **N**, and the parent branch is removed. After determining the dual bound for each branch, the algorithm determines the dual bound for the problem, DB_i, for iteration i. It is defined as $\max\{D_n \;\; \forall n \in \mathbf{N}\}$, and $\mathbf{Q}$ is the set of fixed decisions corresponding to the dual bound of the problem, DB_i.

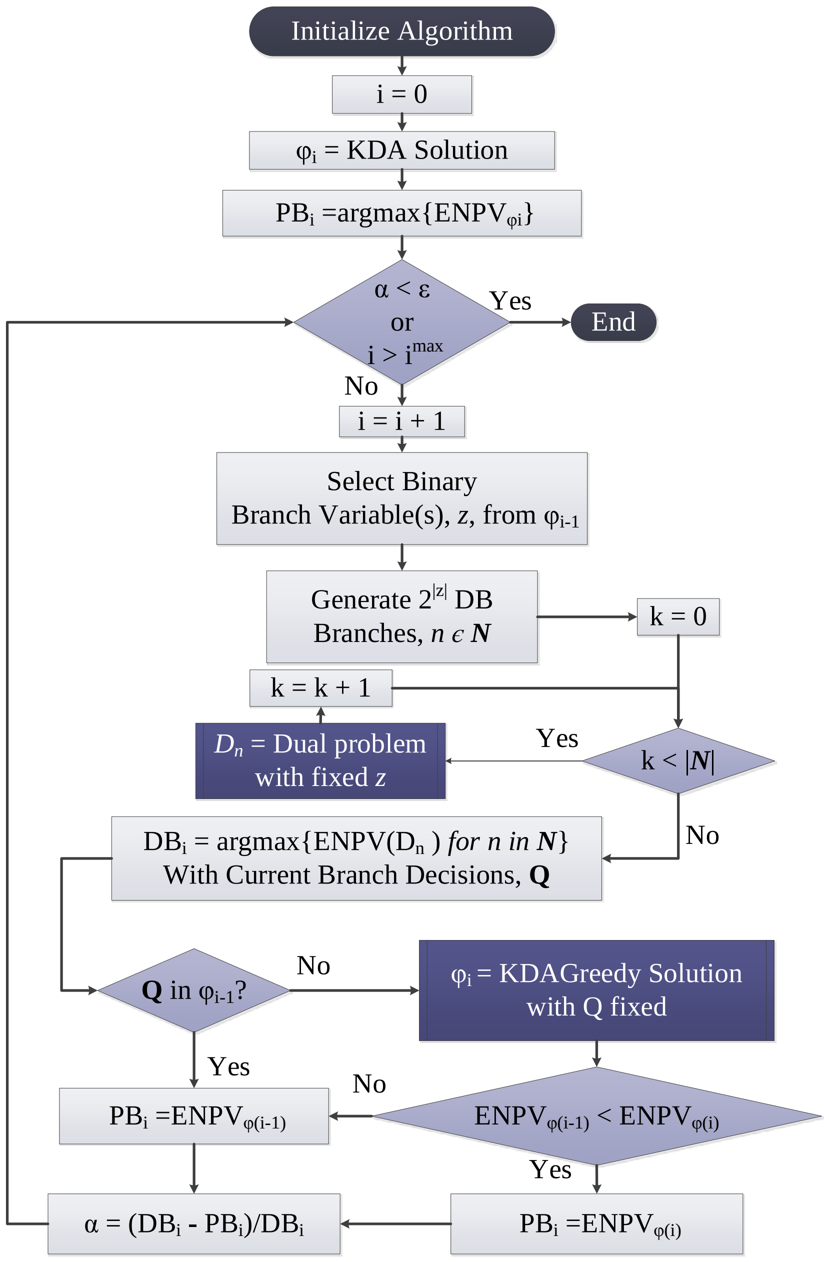

Initialize Algorithm
i = 0
φi = KDA Solution
PBi =argmax{ENPVφi}
α < ε or i > imax
Yes
End
No
i = i + 1
Select Binary Branch Variable(s), z, from φi-1
Generate 2|z| DB Branches, n ϵ N
k = 0
k = k + 1
Dn = Dual problem with fixed z
Yes
k < |N|
No
DBi = argmax{ENPV(Dn) for n in N} With Current Branch Decisions, Q
Q in φi-1?
No
φi = KDAGreedy Solution with Q fixed
Yes
No
PBi =ENPVφ(i-1)
ENPVφ(i-1) < ENPVφ(i)
Yes
α = (DBi - PBi)/DBi
PBi =ENPVφ(i)

Figure 6.1 The branch and bound algorithm

The algorithm continues by comparing the decisions in **Q** with values of the decision variables in the KDA solution, φ_{i-1}. If the values of the decision variables match then the primal bound of the problem, PB_i, is equal to the primal bound of the previous iteration, PB_{i-1}. Otherwise, the algorithm tries to update the primal bound by solving the MSSP where the values of the decision variables in **Q** are fixed using the GreedyKDA (Section 6.2.2). A new feasible solution to the problem, φ_i, is generated. If the EENPV of the new feasible solution (φ_i) is greater than the previous iteration primal bound (PB_{i-1}), the primal bound, PB_i, takes the value of the EENPV of the new feasible solution (φ_i). If the EENPV of the KDA solution is lower than previous iteration primal bound (PB_{i-1}), the value of primal bound, PB_i, is set equal to PB_{i-1}. The algorithm updates the relative gap, α, using the primal and dual bounds of the current iteration, PB_i and DB_i. If the gap is lower than the tolerance, ε, or the maximum iteration count, i^{max}, is reached the algorithm terminates. Otherwise, the algorithm increments the iteration count and selects new branching variables. At termination, the feasible solution at current iteration, φ_i, provides the solution of the MSSP at a relative gap of α.

6.2 Generation of Dual Bounds

6.2.1 A Progressive Hedging (PH) Bounding Approach

he first approach for generating dual bounds solves linear programming (LP) relaxations of a modified version of the original MSSP formulation using the Progressive Hedging (PH) approach originally presented in Rockafellar and Wets[13] and adapted by Watson and Woodruff[15]. The PH approach decomposes the MSSP into individual scenario quadratic programs (QPs) with a modified objective function, and uses the solutions of these QPs to converge to the MSSP solution. This scenario-wise approach allows solving linear MSSPs with exogenous uncertainty without generating the full MSSP.

The SP formulation with exogenous uncertainty can be written using Eqs. 6.1-6.5,

$$\mathbf{min} \sum_{s} p_s \left[\sum_{t \in \mathcal{T}} \left(C'_{t,s} \chi_{t,s} + f'_{t,s} \gamma_{t,s} \right) \right] \tag{6.1}$$

$$\sum_{t' \in \mathcal{T}:\ t' < t} \left(\mathcal{A}_{t',t,s} \chi_{t',s} + \mathcal{B}_{t',t,s} \gamma_{t',s} \right) \leq \mathcal{H}_{t,s} \quad \forall s \in \mathbb{S}, \forall t \in \mathcal{T} \tag{6.2}$$

$$\chi_{1,r} = \chi_{1,s} \quad \forall r, s \in \mathbb{S} : r > s \tag{6.3}$$

$$\begin{bmatrix} \chi_{t+1,r} = \chi_{t+1,s} \\ \gamma_{t,r} = \gamma_{t,s} \end{bmatrix} \quad \forall r, s \in \mathbb{S} : r > s \quad \forall t \in \mathcal{T} : t \leq \boldsymbol{\tau}(r, s) \tag{6.4}$$

$$\gamma_{t,s}\, , \chi_{i,t,s} \in \mathbb{R} \quad \forall t \in \mathcal{T}, (r, s) \in \mathbb{S} \tag{6.5}$$

where $C_{t,s}$ is the cost associated with the decision vector $\chi_{t,s}$, p_s is the probability that the scenario s will occur, and $f_{t,s}$ represents the cost of the scenario specific decisions $y_{t,s}$. Equation 6.2 shows scenarios specific constraints which limit the value decision variables $\chi_{t,s}$ and $y_{t,s}$ can take. Non-anticipativity is enforced using Eqs. 6.3 and 6.4, where Eq. 6.3 ensures that first time period decisions are identical and Eq. 6.4 enforces that decision variable values must be identical until the time period in which the scenarios become differentiable, $\boldsymbol{\tau}(r, s)$. The MSSP formulation given in Eqs. 6.1-6.5 can be converted to its implicit form as shown in Eqs 6.6-6.8.

$$\mathbf{min} \sum_{s} p_s \left[\sum_{t \in \mathcal{T}} \left(C_t x_t + f_t y_{t,s} \right) \right] \tag{6.6}$$

$$\sum_{t' \in \mathcal{T}:\ t' < t} \left(A_{t',t} x_{t'} + B_{t',t,s} y_{t',s} \right) \leq H_{t,s} \quad \forall t \in \mathcal{T} \tag{6.7}$$

$$y_{t,s}\, , x_t \in \mathbb{R} \quad \forall t \in \mathcal{T}, (r, s) \in \mathbb{S} \tag{6.8}$$

The PH algorithm used in this work is given in Fig. 6.2,which follows the notation given in Watson and Woodruff[15]. The first step is to initialize the iteration counter (k) to zero, the algorithm then solves the deterministic optimization problem for each scenario, finding $x^{(k)}$ (Fig. 6.2, Step 2). The construction of the deterministic optimization problem is done using the implicit form of the MSSP shown in Eqs. 6.6-6.8. Next, the average values for the decision vector, $\bar{\boldsymbol{x}}^{(k)}$, and the weights, $w_s^{(k)}$, are calculated (Fig. 2, Steps 3 and 4). In Step 5, the iteration counter is incremented, and new QPs are constructed using the values of average values for the decision vector, $\bar{\boldsymbol{x}}^{(k)}$**,** and the weights, $w_s^{(k)}$. Solutions of these QPs are used to update the average values and weights (Fig. 6.2, Steps 7 and 8). The convergence of the algorithm is checked in Step 10 using the value of $g^{(k)}$ calculated in Step 9 (Fig. 6.2). If $g^{(k)}$ is below a pre-set tolerance value, $\mathcal{E}$, it terminates. Otherwise, it returns to Step 5.

```
1.  $k := 0$
2.  $For\ all\ s \in S, x_s{}^{(k)} := argmin(cx + f_s y_s): (x, y_s) \in Q_s$
3.  $\bar{x}^{(k)} := \sum_{s \in S} \Pr(s) x_s{}^{(k)}$
4.  $w_s{}^{(k)} := \rho\left(x_s{}^{(k)} - \bar{x}^{(k)}\right)$
5.  $k := k + 1$
6.  $For\ all\ s \in S,$
7.  $x_s{}^{(k)} := argmin\left(cx + w_s{}^{(k-1)}x + {}^{\rho}/_{2} \left\|x - \bar{x}^{(k)}\right\|^2 + f_s y_s\right) : (x, y_s) \in Q_s$
8.  $\bar{x}^{(k)} := \sum_{s \in S} \Pr(s) x_s{}^{(k)}$
9.  $For\ all\ s \in S, w_s{}^{(k)} := w_s{}^{(k-1)} + \rho\left(x_s{}^{(k)} - \bar{x}^{(k)}\right)$
10. $g^{(k)} := \sum_{s \in S} \Pr(s) \left\|x - \bar{x}^{(k)}\right\|$
11. $If\ g^{(k)} < \mathcal{E}, then\ go\ to\ 5. Otherwise\ terminate$
```

Figure 6.2. The progressive hedging algorithm (Watson & Woodruff, 2011)

Progressive hedging approach has been proven to converge to the optimal solution for convex multistage stochastic programs with exogenous uncertainty and continuous decision variables[15]. The problems of interest in this work are MSSPs with integer decision variables and endogenous uncertainty, where the differentiating events can be specified but not when and if they would occur. To ensure that the solutions obtained by the PH algorithm are true dual bounds for the MSSPs with integer decision variables and endogenous uncertainty, the integrality constraints of the MSSP are relaxed, and appropriate upper and lower bounds for these variables are introduced. Relaxing the integrality constraints does not significantly change the difficulty of solving the full space model. The difficulty in solving the full space model arises from number of variables and the interconnection between the variables. Relaxation of the NACs can simplify problem however it cannot address the growth in the number of variables associated with the increase in the number of scenarios. Despite having a more complex objective function, the PH algorithm is capable of reducing the complexity of the relaxed MSSP by separating the problem into scenario sub-problems.

The PH algorithm requires the knowledge of the scenarios and their differentiation time periods. When handling endogenous uncertainty the time period of differentiation is not known *a priori*. The endogenous uncertain parameters considered in this work are of Type II[14], where only the timing of the realization is impacted by the value of the decision variables. As such, the relaxed MSSPs can be converted to two-stage linear SP formulations that resemble linear SP models with only exogenous uncertain parameters by removing all but the current-stage NACs and by reducing the current-stage NACs to a subset of NACs where the time period of differentiation is "known".

In the formulation of the MSSP with endogenous uncertainty, NACs are conditionally enforced based on the value of the indicator variable. To convert the conditional NACs to the form shown in Eq. 6.4, we start by identifying the subset of current-stage NACs which do not require any additional decisions to be taken for realizing their outcome. Note that for this subset of current-stage NACs, the time period of differentiation is known, therefore their NACs can be written similar to Eq. 6.4 until the earliest time period at which the scenarios can become differentiable. After identifying this subset of current-stage NACs, a set $\boldsymbol{\tau}'(i, r, s)$ is constructed. The set $\boldsymbol{\tau}'(i, r, s)$ contains the earliest time period at which scenarios r and s can become differentiable with respect to endogenous uncertain parameter $i \in \mathcal{J} \in \mathbb{I}$ (i.e. the time earliest time period at which $Z_{t,r,s}$ can become equal to one.). This allows for the construction of Eqs. 6.9-6.13 which are identical in form to Eqs. 6.1-6.5 but include endogenous uncertain parameters.

$$\min \sum_{s} p_s \left[\sum_{t \in \mathcal{T}} \left(\sum_{i \in \mathbb{I}} \left(D_{i,t} b_{i,t,s} \right) + f_{t,s} \gamma_{t,s} \right) \right] \tag{6.9}$$

$$\sum_{t' \in \mathcal{T}:\ t' < t} \left(\sum_{i \in \mathbb{I}} \left(G_{i,t,t',s} b_{i,t,s} \right) + B_{t',t,s} \gamma_{t',s} \right) \leq a_{t,s} \quad \forall s \in \mathbb{S}, \forall t \in \mathcal{T} \tag{6.10}$$

$$b_{i,1,r} = b_{i,1,s} \quad \forall r, s \in \mathbb{S}, \forall i \in \mathbb{I} \tag{6.11}$$

$$\begin{bmatrix} b_{t+1,r} = b_{t+1,s} \\ \gamma_{t,r} = \gamma_{t,s} \end{bmatrix} \quad \forall r, s \in \mathbb{S} : r > s \quad \forall t \in \mathcal{T} : t \leq \min\{\boldsymbol{\tau}'(i,r,s)\ \ \forall\, i \in \mathcal{I}\} \tag{6.12}$$

$$\gamma_{t,s}\,, b_{i,t,s} \in \mathbb{R} \quad \forall t \in \mathcal{T}, (r,s) \in \mathbb{S} \tag{6.13}$$

The solutions obtained by the PH algorithm (Fig. 6.2) to these relaxed two-stage linear SPs are used to update the dual bounds of the MSSP (Fig. 6.1) at each iteration. We will refer to the branch-and-bound algorithm with progressive hedging approach as PH-KDA.

6.2.2 An Optimal Scenario Solution (OSS) Bounding Approach

The second bounding approach for generating dual bounds solves the original MSSP without enforcing any non-anticipativity constraints, practically decomposing the MSSP into individual scenario sub-problems. The optimum solutions of individual scenario sub-problems provide a weaker bound than the PH approach, however, unlike PH, the dual bound is obtained without multiple iterations for convergence. This approach may be particularly useful when the size of the scenario set is prohibitively large.

Implementing the procedure starts by removing all non-anticipativity constraints from the original MSSP. The resulting formulation is,

$$\min \sum_{s} p_s (C x_s + f_s y_s)$$

$$s.t. (x_s, y_s) \in Q_s \; \forall \, s \in S$$

where the objective function is simply the probability weighted cost of each scenario. It should be noted that the formulation assumes that the constraint set, Q_s, is separable and that the integrality constraints are not relaxed. Furthermore, the problem can be separated such that the value of the objective function for the original problem is given as the probability weighted Optimal Solution of each Scenario sub-problem (OSS). Computing the optimal solution to each scenario sub-problem is highly parallelizable and considerably less complex than solving the original MSSP problem. We will refer to the branch-and-bound algorithm that uses optimal solution of individual scenario problems for generating dual bounds as OSS-KDA.

6.3 Generating Feasible Solutions and Primal Bounds Using the GreedyKDA

The primal bound of the problem is updated using a modified version of the KDA[12]. The original KDA uses a series of knapsack problems to find a feasible solution for MSSPs with endogenous uncertainty. The KDA begins at $t = 0$ by decomposing each of the decision variables $b_{i,t,s}$ into a set of items. Each item has an associated value. The value for each item is calculated as the contribution of the item's corresponding decision variable to the objective function. If the value of the item depends on the underlying value of an uncertain parameter, the expected value of the uncertain parameter is used to calculate the value. The knapsack problem also has a set of weight constraints. The weight constraints in the KDA are composed of constraints (e.g., available resources) from the original MSSP which limit the number of items that can be packed in the knapsack.

The KDA starts by packing an initial knapsack. The knapsack problem is used to determine which items are packed and the fraction of the item packed if items represent continuous variables. The selected items and their fractional values are used to determine the value of the decision variables at the first time period in the planning horizon. The uncertainty associated with those decision variables are realized, and the KDA generates a new knapsack problem for each realization. Based on the realizations, the algorithm decides which items are eligible to be considered in each of the newly created knapsack problems, and solves them. Solution of each knapsack problem determines the values of the decision variables, which in turn results in realizations of associated uncertain parameters. The KDA continues until the end of the planning horizon.

In some instances it may be beneficial to introduce additional weight constraints to the knapsack problems solved by the KDA to direct its solution away from the greedy one. These constraints, which we refer as heuristic overscheduling constraints, may help KDA to avoid over-utilizing resources early in the planning horizon by preventing the selection of items where there are not sufficient resources for investment in future.[12] A full, in depth description of the KDA applied to clinical trial planning can be found in Christian and Cremaschi[12].

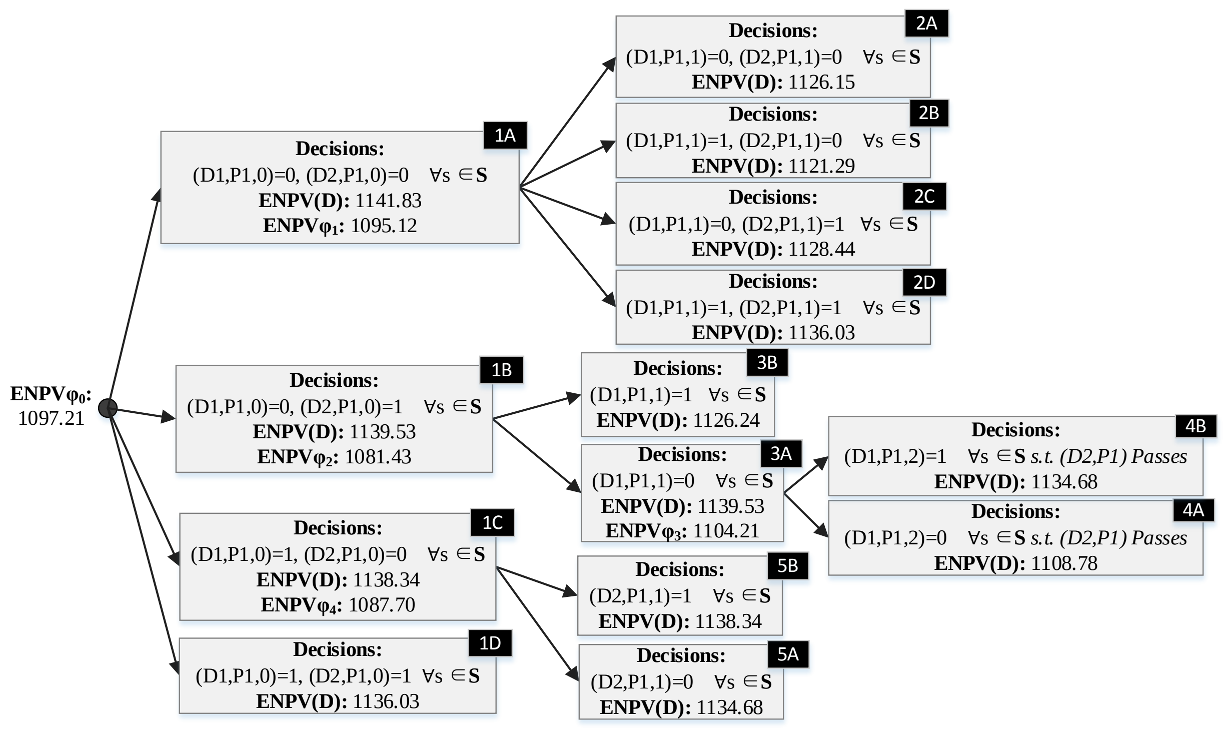
Primal bound updates are constructed using a version of the KDA, GreedyKDA, where no additional weight constraints are added. Without the heuristic overscheduling constraints, the GreedyKDA is able to pack any eligible item in any knapsack while keeping the feasibility of its solution. In the original KDA, new knapsacks are only generated after all uncertainty associated with selected items was realized. In the GreedyKDA, new knapsack problems are generated at every time period allowing non-zero decision variable values if there are enough resources at any time period.

6.4 Case Studies

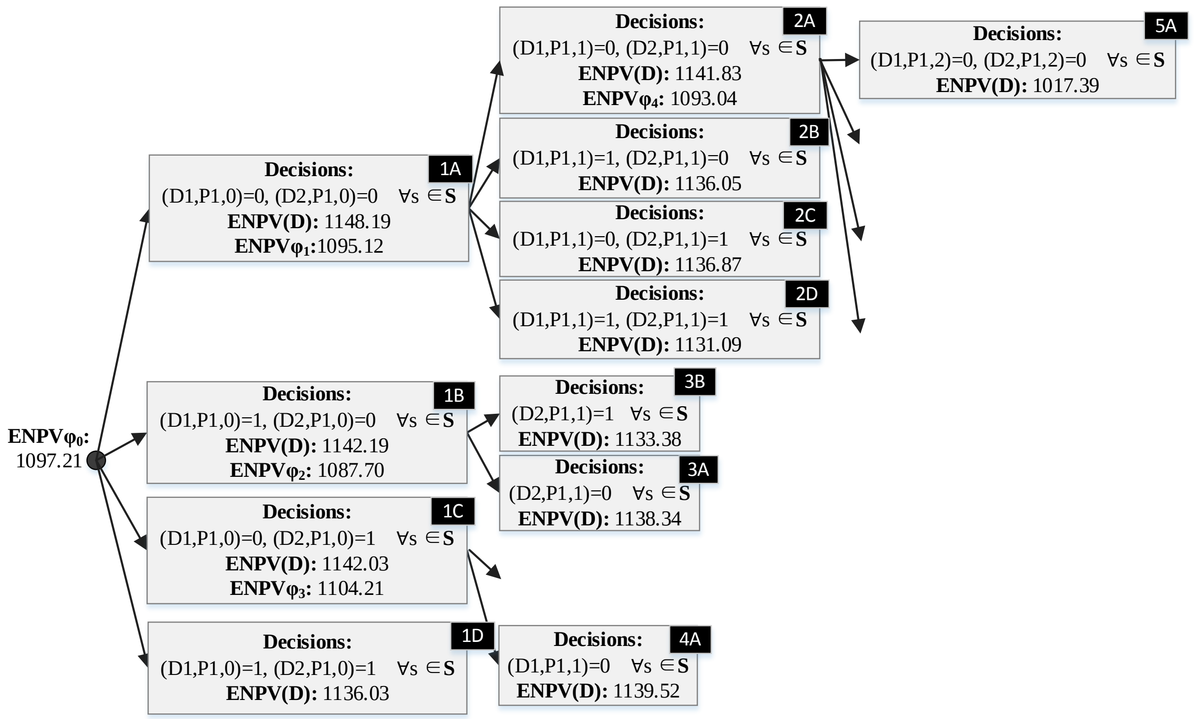
We use the branch and bound algorithm to solve several instances of the pharmaceutical R&D pipeline clinical trial planning problem. Both PH-KDA and OSS-KDA are used to solve a small two-product case. The two-product case has two instances '2P2T' and '2P3T', which represent two products (2P) with two clinical trials (2T) and two products (2P) with three clinical trials (3T). We also consider a three product instance (3P3T), a four-product instance (4P3T), and a five-product instance (5P3T) all with three clinical trials. In the case of the OSS-KDA, preliminary results prompted us to also solve six-product (6P3T) and seven-product (7P3T) case studies. A brief overview of the clinical trial planning problem is provided in the section below. Parameter values for each case can be found in Christian and Cremaschi[12] and Christian and Cremaschi[16]. The branch and bound algorithm is implemented in Python 3.5. Both dual bounding approaches and the GreedyKDA utilize Pyomo 4.1[17] and CPLEX 12.6. The solutions to the deterministic equivalent MSSPs for all cases were found using Pyomo 4.1 and CPLEX 12.6. All of the problems in this work were solved using the Auburn University Hopper Cluster.

6.5 Results and Discussion

The branch and bound algorithm is first demonstrated using a toy-box sized two-product two-clinical-trial (2P2T) case. We present the first five iterations of each algorithm in Fig. 6.3. The initial primal bound obtained by KDA is 1097. The algorithm uses the solution from the initial primal bound to select decisions variables to branch on. In this problem, the algorithm selects starting first clinical trials of both Drug 1 and Drug 2 at first time period, i.e., (D1, P1, 0) and (D2, P1,0). Selecting two decision variables to branch on creates four branches marked as 1A, 1B, 1C, and 1D in Fig. 3. In case of PH-KDA, the LPs are generated, and their solutions – obtained by the PH algorithm – are 1141.83 (1A – Fig 6.3(A)), 1139.53 (1B – Fig 6.3(A)), 1138.34 (1C – Fig 6.3(A)), and 1136.03 (1D – Fig 6.3(A)). In case of OSS-KDA, the individual scenario mixed integer linear programs (MILPs) are generated, and their solutions are 1148.19 (1A – Fig 6.3(B)), 1142.19 (1B – Fig 6.3(B)), 1142.03 (1C – Fig 6.3(B)), 1136.03 (1D – Fig 6.3(B)). At first iteration, the dual bounds are 1141.83 and 1148.19 for PH-KDA and OSS-KDA, which yield relative bounds of 0.041 and 0.047, respectively (Table 6.1).


ENPVφ0: 1097.21
1A Decisions: (D1,P1,0)=0, (D2,P1,0)=0 ∀s ∈S ENPV(D): 1141.83 ENPVφ1: 1095.12
1B Decisions: (D1,P1,0)=0, (D2,P1,0)=1 ∀s ∈S ENPV(D): 1139.53 ENPVφ2: 1081.43
1C Decisions: (D1,P1,0)=1, (D2,P1,0)=0 ∀s ∈S ENPV(D): 1138.34 ENPVφ4: 1087.70
1D Decisions: (D1,P1,0)=1, (D2,P1,0)=1 ∀s ∈S ENPV(D): 1136.03
2A Decisions: (D1,P1,1)=0, (D2,P1,1)=0 ∀s ∈S ENPV(D): 1126.15
2B Decisions: (D1,P1,1)=1, (D2,P1,1)=0 ∀s ∈S ENPV(D): 1121.29
2C Decisions: (D1,P1,1)=0, (D2,P1,1)=1 ∀s ∈S ENPV(D): 1128.44
2D Decisions: (D1,P1,1)=1, (D2,P1,1)=1 ∀s ∈S ENPV(D): 1136.03
3B Decisions: (D1,P1,1)=1 ∀s ∈S ENPV(D): 1126.24
3A Decisions: (D1,P1,1)=0 ∀s ∈S ENPV(D): 1139.53 ENPVφ3: 1104.21
4B Decisions: (D1,P1,2)=1 ∀s ∈S s.t. (D2,P1) Passes ENPV(D): 1134.68
4A Decisions: (D1,P1,2)=0 ∀s ∈S s.t. (D2,P1) Passes ENPV(D): 1108.78
5B Decisions: (D2,P1,1)=1 ∀s ∈S ENPV(D): 1138.34
5A Decisions: (D2,P1,1)=0 ∀s ∈S ENPV(D): 1134.68

(A)


ENPVφ0: 1097.21
1A Decisions: (D1,P1,0)=0, (D2,P1,0)=0 ∀s ∈S ENPV(D): 1148.19 ENPVφ1:1095.12
1B Decisions: (D1,P1,0)=1, (D2,P1,0)=0 ∀s ∈S ENPV(D): 1142.19 ENPVφ2: 1087.70
1C Decisions: (D1,P1,0)=0, (D2,P1,0)=1 ∀s ∈S ENPV(D): 1142.03 ENPVφ3: 1104.21
1D Decisions: (D1,P1,0)=1, (D2,P1,0)=1 ∀s ∈S ENPV(D): 1136.03
2A Decisions: (D1,P1,1)=0, (D2,P1,1)=0 ∀s ∈S ENPV(D): 1141.83 ENPVφ4: 1093.04
2B Decisions: (D1,P1,1)=1, (D2,P1,1)=0 ∀s ∈S ENPV(D): 1136.05
2C Decisions: (D1,P1,1)=0, (D2,P1,1)=1 ∀s ∈S ENPV(D): 1136.87
2D Decisions: (D1,P1,1)=1, (D2,P1,1)=1 ∀s ∈S ENPV(D): 1131.09
5A Decisions: (D1,P1,2)=0, (D2,P1,2)=0 ∀s ∈S ENPV(D): 1017.39
3B Decisions: (D2,P1,1)=1 ∀s ∈S ENPV(D): 1133.38
3A Decisions: (D2,P1,1)=0 ∀s ∈S ENPV(D): 1138.34
4A Decisions: (D1,P1,1)=0 ∀s ∈S ENPV(D): 1139.52

(B)

Figure 6.3. Five iterations of the branch and bound algorithm for the two-product two-clinical trial case study for (A) the PH-KDA and (B) OSS-KDA

Because the upper bound of 1A is the highest, the algorithm sets the values of the decision variables associated with (D1, P1, 0) and (D2, P1,0) equal to zero, and calls GreedyKDA. The solution obtained by the GreedyKDA suggests starting the first clinical trials of D1 and D2 at the second time period (*t*=1). The algorithm generates four new branches (2A-D), and finds the dual bounds. The algorithm continues by selecting the end branch with the highest upper bound (e.g, 1B for the third iteration) until the stopping criteria are met. The progression of the primal bound, dual bound, and relative gap for each iteration shown in Fig. 6.3 are tabulated in Table 6.1.

Table 6.1 Values for the primal bound, dual bound, and relative gap for the first five iterations of the PH-KDA and OSS-KSA shown in Figure 6.3.

		PH-KDA			**OSS-KDA**	
Iteration	PB	DB	Gap	PB	DB	Gap
1	1097.21	1141.83	0.041	1097.21	1148.19	0.047
2	1097.21	1139.53	0.039	1097.21	1142.19	0.041
3	1104.21	1139.53	0.032	1097.21	1142.03	0.041
4	1104.21	1138.35	0.031	1097.21	1141.83	0.041
5	1104.21	1138.35	0.031	1104.21	1139.52	0.032

We tested PH-KDA on a two-product, three clinical trial (2P3T) and a three product, three clinical trial (3P3T) case. Figure 6.4 plots the log (base 10) of the CPU time consumed by the PH-KDA versus the relative gap ((DB_i-PB_i)/DB_i) for the two- and three-product cases. Labels on the marker on the graph identify the number of completed iterations. In all three cases, the slopes in Fig. 6.4 are approximately linear indicating a logarithmic relationship between the CPU time and the relative gap. Therefore, the decision variables branched on in earlier iterations have a larger impact on the relative gap then the decisions branched on in later iterations, which initially improves the dual bound rapidly. The quality of the PH dual bound is limited due to the linear relaxation of the MSSP and use of only a subset of next stage NACs. For the two-product two-trial (2P2T) case, the algorithm terminated when relative gap was below the specified tolerance of 0.5% at the 44nd iteration. In contrast, for the two-product three-trial (2P3T) case, the algorithm was able reduce the relative gap to 1.7% in 50 iterations and terminated. Premature termination of the algorithm in the two-product three-trial case was also caused by the PH dual bound. To ensure that the PH algorithm solution provides a true dual bound for the problem, all integrality constraints are relaxed. Hence, the solutions at the dual bound allows partial investments (i.e. non-integer results) on some of the clinical trials. The GreedyKDA only generates feasible solutions, in which these non-integer decision variables become zero and are never branched on. For the three-product three trial (3P3T) case study, the algorithm ran for the maximum allowable wall time of ten days. At termination, the relative gap was 2.6%. The time needed to run each algorithm to completion is shown in Table 6.2. As expected, compared to the deterministic equivalent MSSP (also in Table 6.2), the PH-KDA takes significantly longer to close the gap. The MSSPs for the case studies considered in this work are quick to solve however, it should be noted that the time and memory required to solve MSSPs grows exponentially which makes the development of case studies which have MSSPs which sufficiently complex and still solvable very difficult. For completeness we included case studies which span from trivial (2P2T) to unsolvable (7P3T). As the size of problem increases, the time of convergence of the PH algorithm increases. This results in very slow convergence of the PH-KDA for large instances of the clinical trial planning problem (i.e. six or more products). In the six-product three clinical trial case, the PH-KDA was only able to complete a few iterations. It should be noted that the first iteration of PH-KDA provide a feasible solution and its quality (via the first dual bound) for the original MSSP. The algorithm generated an initial relative gap of 6.8% for the six-product case.

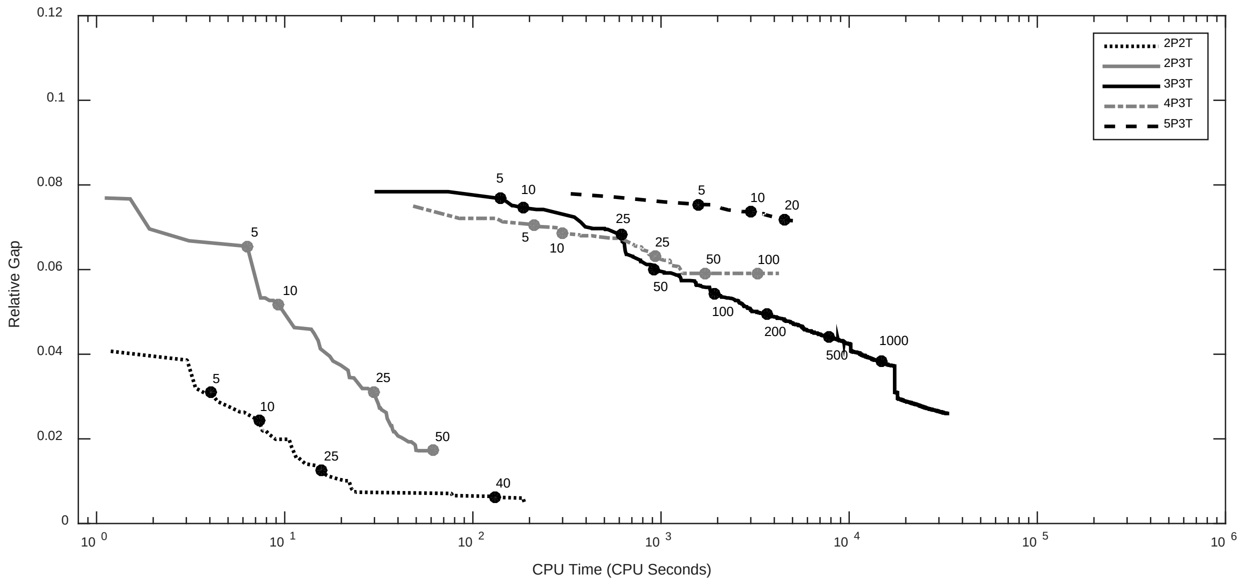


Figure 6.4. Plot of CPU time vs. relative gap for the pharmaceutical clinical trial planning case studies for the PH-KDA algorithm

Figure 6.5 shows the log (base 10) of the CPU time vs the relative gap for OSS-KDA algorithm on three clinical trial two- (2P3T), three- (3P3T), four- (4P3T), five- (5P3T), six-(6P3T), and seven- (7P3T) product case studies in addition to the two-product two clinical trial (2P2T) instance. Comparing the performances of the PH-KDA (Figure 6.4) and the OSS-KDA (Figure 6.5), it can be seen that the PH-KDA provides a tighter initial dual bound, and a smaller relative gap, than the OSS-KDA. The tighter initial bound on the problem suggests that the subset of current-stage NACs in the PH bounding approach play a non-trivial role in the quality of the dual bound. The slope of the lines in Fig. 6.5 are less steep than the lines in Fig. 6.4. The steepness of the lines is an indicator that the bound improvement is faster when for the PH-KDA where the problem includes a subset of the NACs. Unlike the PH-KDA, the OSS-KDA converges to the optimal solution for both the two-product, two-trial (2P2T) instance and the two-product, three-trial (2P3T) instance. This is because the OSS-KDA does not require relaxation of the integrality constraints. From Fig. 6.5, we can conclude that given sufficient time, the algorithm would eventually converge to the optimal solution.

Applying the OSS-KDA to the seven-product clinical trial planning problem generated an initial relative gap of 10%; after 200 iterations, the gap was reduced to 8%. To date, we have been unable to generate and solve the deterministic equivalent MSSP of the seven-product clinical trial planning problem. The problem has 16,384 scenarios, more than 11 million variables, and more than 39 million constraints. Generating the problem requires a random-access memory (RAM) more than 126 GB, and CPLEX 12.63 was not able to solve the problem on a node with sixteen cores and 256 GB of RAM. Based on information collected from the solution log files, CPLEX is unable to complete the pre-solve routine for the seven product case.

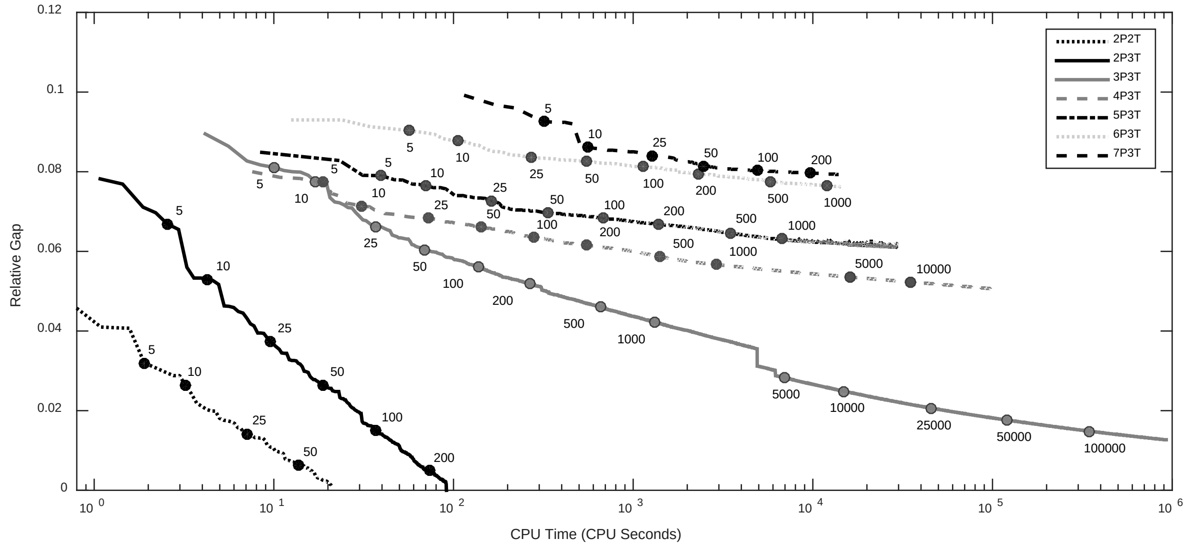


Figure 6.5. Plot of CPU time vs. relative gap for the two-, three-, five-, six-, and seven-product cases using OSS-KDA.

For both PH-KDA and OSS-KDA, the branch and bound algorithm closes the gap quicker at the early iterations and its convergence slows down with the number of iterations because both approaches branch on more decision variables early on. Because fewer scenarios are differentiable at the earlier iterations, the algorithm branches on a larger number decision variables, and fixing the values of these decision variables, in general, makes a significant impact on the gap. As the algorithm continues, more scenarios become differentiable and the number of variables branched on at each iteration decreases. It is worth noting that the convergence of the PH-KDA cannot be guaranteed due to the convergence criteria of the PH approach (i.e., the convergence of PH-KDA may not converge at every iteration. Unlike PH-KDA, the OSS-KDA is guaranteed to converge to the optimum given sufficient time. The branch and bound algorithm only closes a branch whose dual bound is lower than that of the primal bound, overall convergence of the algorithm is quite slow. As the number of iterations increases, the optimality gap for larger case studies (Fig. 6.5) will continue to slowly decrease as decision variables (although at lower numbers of them) are fixed and dual bound tightens. Then, there will be sudden drops (of the optimality gap) at time periods where fixing the decision variables improves the primal bound. We would also expect for the drops to gradually become smaller and farther apart because the algorithm branches on fewer variables with each iteration similar to the PH-KDA.

One of the challenges with solving real-world size MSSPs is the space complexity of the problem (i.e. the RAM required to generate the problem). As can be seen from Table 6.2, the PH-KDA and OSS-KDA use significantly less RAM than its deterministic equivalent counterpart for most case studies. The RAM usage of the both the PH-KDA and the OSS-KDA are higher for the three product case that the deterministic algorithm. This is caused by the nature of the algorithms. The primary source of memory usage in the branch and bound algorithms is in the storage of branches. The storage of these branches gradually increase memory requirements of the algorithm. In this implementation of the algorithm, the information for each branch is stored in RAM. Storing branch information in RAM is not required and memory usage would be improved if branch information was databased. In both the OSS-KDA and the PH-KDA the 3P3T problem required more memory than the 5P3T problem. This is caused by the number of iterations completed. In both cases, the 3P3T case completes significantly more iterations which results in a larger number of branches being stored.

Table 6.2 also shows a large difference between the OSS-KDA and the PH-KDA in terms of the amount of time it takes to solve the problem. In even the smallest cases OSS-KDA provides results faster than the PH-KDA. This is caused by the nature of the dual bounding approaches. In the PH approach bounds are found using a scenario-based iterative scheme whereas the OSS, also a scenario-based scheme, solves in a single iteration. This suggests with the current implementation, the trade-off for a better initial bound with PH-KDA may not be worth the additional computation time.

Table 6.2 Resource usage, relative gap, and computational time results for the deterministic equivalent MSSP and the branch and bound algorithm

	Deterministic			PH-KDA			OSS-KDA		
	RAM (MB)	Relative Gap	CPU Time	RAM (MB)	Relative Gap	CPU Time	RAM (MB)	Relative Gap	CPU Time (HH:MM:SS)
2P2T	2.45	0.001	0:00:01	0.05	0.005	0:03:08	0.00	0.009	0:00:21
2P3T	5.93	0.001	0:00:01	0.48	0.017	0:01:01	0.04	0.000	0:01:31
3P3T	89.79	0.001	0:00:03	26.07	0.026	615:07:05	174.70	0.013	257:30:18
5P3T	1430.15	0.001	0:00:42	2.12	0.068	643:41:52	60.02	0.068	8:10:17

The impact of parallelizing the PH algorithm of PH-KDA is studied for the five-product three-clinical trial (5P3T) case study. The problem has a total of 1024 scenarios. Table 6.3 shows the number of threads used for parallelization for each instance of the problem along with the approximate number of problems solved per thread per PH iteration. Because CPLEX 12.6 recommends allocating additional processor cores for solution of the QPs when available, six cores were allocated for each thread. This also improved the efficiency of the PH algorithm. Based on the number of iterations completed, parallelization has the greatest impact when nine threads are used. In the case where nine threads were used, 183 iterations were completed and the relative gap of the problem was reduced to 6.4%. However, our limited computational experiments showed that increasing from three threads to six threads had an inverse effect on the solution quality. The impact of parallelizing the OSS-KDA is straight forward. Since the OSS scenario MILP sub-problems are trivial, it is sufficient to say that sub-problems should be divided evenly between available cores unless the number of sub-problems solved on each core is not sufficient to outweigh the additional time required to parallelize the algorithm.

Table 6.3 The Impact of running PH algorithm of PH-KDA in parallel for the five-product case study

Number of Threads	Completed Iterations	Relative Gap	Problems Per Thread
3	39	0.0684	341
6	41	0.0772	170
9	183	0.064	113

The current implementation of the branch and bound algorithm selects decisions to branch on from the KDA decision tree. In general, the algorithm selects decision variables from the GreedyKDA solution that have non-zero values starting with the variables earliest in the planning horizon. When the algorithm selects which decision variables to branch on, it selects all the decisions in one particular branch of the KDA decision tree. Because the object that holds the decision tree is not ordered, the algorithm may not always select the same branch (i.e., corresponding to the same realizations) from the KDA decision tree. The performance of the algorithm when the parallelization studies were conducted suggests that it is particularly sensitive to the order in which decision variables are selected for branching.

6.6 Conclusions

In this work, we introduced a branch and bound algorithm for solving multistage stochastic programs (MSSPs) with endogenous uncertainty. The algorithm uses our novel Knapsack-problem based Decomposition Algorithm (KDA) to efficiently generate feasible solutions and primal bounds for the MSSPs. We investigated two approached to generate dual bounds. The first one generated two-stage linear stochastic programs by relaxing the integrality constraints of the original MSSP and removing all but the current-stage non-anticipativity constraints (NACs) and by reducing the current-stage NACs to a subset of NACs where the time period of differentiation is "known". Then, these two-stage linear stochastic programs are solved using used progressive hedging (PH). The second approach solved the individual scenario sub-problems to optimality, and estimated and updated the dual bound as probability weighted Optimal Solution of each Scenario sub-problem (OSS). Both implementations of the branch and bound algorithm were applied to solve several instances of the pharmaceutical clinical trial planning problem. Our studies reveal that, in all case studies and for both implementations, the CPU time for the algorithm is higher than the deterministic MSSP if the deterministic MSSP can be generated and solved with available computational resources. However, using the OSS-KDA, we were able to solve a seven-product instance of the clinical trial planning problem to an 8% relative gap, and we have not been able to solve the deterministic equivalent MSSP with similar computational resources. Despite having a slower convergence time, the first iteration of the algorithm provides a feasible solution, a primal bound, and a dual bound for the problem.

For future work, three paths have been identified to increase the effectiveness and efficiency of the algorithm. First, to address the challenge the integrality constraints not being enforced in the PH-KDA, an additional step will be modified to branch on the variables which take non-integer values and cause the upper bound not to converge to the primal bound. Second, we plan to improve implementation of both the PH-KDA and the OSS-KDA by investigating different rules for selecting the branching variables and further parallelization approaches. One approach for selecting branching variables would be to prioritize branching on decision variables associated with branches in the decision tree where there exist more scenarios. Third, memory management of the algorithm information will be improved by implementing a database structure.

CHAPTER 7
A GRAPH THEORY APPROACH TO NON-ANTICIPATIVITY CONSTRAINT GENERATION IN MULTISTAGE STOCHASTIC PROGRAMS WITH INCOMPLETE SCENARIO SETS

The formulation of a multistage stochastic program requires the equality of decision variables in different scenarios in order to ensure that the realized values of the uncertain parameters are not anticipated. Depending on the type of uncertainty in the problem this can be accomplished either implicitly or explicitly. When the timing of the realization of uncertainty is known (i.e. exogenous uncertainty), anticipation of the underlying value of the uncertain parameters can be prevented by defining a variable used to relate the decision variables in two different scenarios. Using the same decision variable in multiple scenarios ensures that the decision variable values are identical in scenarios where uncertainty has not been realized. Alternatively, a set of equality constraints, called non-anticipativity constraints (NACs), which sets the values of the decision variables equal to each other at time periods before uncertainty is realized can be used. If the time period of differentiation is not known (i.e. endogenous uncertainty), a set of non-anticapitivity constraints must be used as there is no way to implicitly incorporate the realization of uncertainty.

Approaches for generating NACs for endogenous uncertainty have been studied in the literature before. A full set of NACs can be generated by relating each scenario to every other scenario. This produces a very large number of constraints which are often redundant. Several authors have presented properties of MSSPs, which reduce the number of redundant NACs. Applicability of a lot of these properties is limited to MSSPs where the scenario set is considered complete (i.e., given by the Cartesian product) (Goel & Grossmann, 2006).

Often situations arise where the set of scenarios considered in the MSSP is given by a subset of the full set of scenarios. For instance, every scenario in the full set generated by the Cartesian product may not be realizable. Alternatively, a sub-set of the full scenario set may be used in cases where solving the MSSP with the full scenario set is computationally prohibitive. Properties used to generate NACs that are applicable for the full set of scenarios do not necessarily apply to sub-sets of scenarios. Thus, recent advances have focused on the generation of NAC sets for sets of scenarios which are not generated by the Cartesian product. Hooshmand and Mirhissani (2016) developed two approaches; the first one used a mixed integer linear programming formulation (MILP) to find the minimum cardinality NAC set, and the second one employed graph theory, which added and removed NACs until there were no violations of non-anticipativity. Apap and Grossmann (2016) extended the graph-theory approach to handle both endogenous and exogenous uncertainty. Boland et al. (2016) developed a set of proofs which define the conditions required to omit NACs for an incomplete sets from the problem formulation. In all these works, the authors used the concept of a differentiator set, i.e., a set of sets which store information on the realization of uncertainty required for two scenarios to be differentiable. The differentiator set is sufficient if the realization of the uncertain parameter is instantaneous, implying that the underlying value is revealed using a single indicator. If the realization of uncertainty is gradual there may be several indicators which result in partial revelation of uncertainty. The approaches developed to date cannot be applied to generate NACs of incomplete scenario sets for this type of uncertainty.

We introduce and describe in detail a graph-theory based approach that generates a minimum cardinality NAC set for MSSPs with incomplete scenario set where the realization of uncertainty occurs gradually. Section 7.1 introduces the general formulations of MSSP with endogenous and with endogenous and exogenous uncertainty. The formulation for the MSSP with just endogenous uncertainty is given in Section 7.1.1 and the extension to MSSPs with both endogenous and exogenous uncertainty. In Section 7.2, we propose an algorithm which will generate the minimum cardinality non-anticipativity constraint set. The algorithm was tested using several different case studies. Section 7.3.1 provides a visualization of the performance of the algorithm. In section 7.3.2, presents the case studies used to test the algorithm. The results of the computational studies are also presented in Section 7.4. Conclusions and future directions are given in Section 7.5.

7.1 Background

7.1.1 Mathematical Representation of Multistage Stochastic Programs with Endogenous Uncertainty

The presentation of the general form of the MSSP with endogenous uncertainty is derived from Apap and Grossmann (2016) and Hooshmand and Mirhassani (2016). Suppose that we have a multistage stochastic program where the planning horizon is defined as $\mathcal{T} = \{1,2,3,..,T\}$. For simplicity, the remainder of the paper will define sets using the notation $[\mathcal{T}]$ to represent ordered sets. We first define a set of sources of endogenous uncertainties, $i \in \mathbb{I}$, and a vector of uncertain parameters, θ_i, associated with uncertainty source i. The realizable values of the uncertain parameter, θ_i, are represented using a finite set $\Theta_i = \{\theta_i^1, \theta_i^2, \dots, \theta_i^{\mathcal{R}}\}$. The full scenario set is constructed using the Cartesian product, namely, the scenario set is defined as, $\mathbb{S} := \times_{i\in\mathbb{I}}\{\theta_i\}$. The number of scenarios can be calculated using the cardinality of Θ_i, where $|\mathbb{S}| = \prod_{i\in\mathbb{I}} |\Theta_i|$.

In the problem, we define two decision variables, $b_{i,t,s}$ and $y_{t,s}$, the endogenous stage decision variables and recourse action variables at each time period, $t \in [\mathcal{T}]$, and for each scenario, $s \in \mathbb{S}$. The model EN which represents the standard formulation for the MSSP with endogenous uncertainty is composed of five major parts. The objective function (Eq. 7.1) minimizes the expected value of the total cost of the decision variables. Constant $D_{i,t}$ and $f_{t,s}$ represent the cost associated with decision variables $b_{i,t,s}$ and the cost of recourse decision variable $y_{t,s}$. Equation 7.2 represents the second component of the MSSP, a set of scenario specific constraints. The constraints are functions of the scenario specific decision variables and constants

$G_{i,t,t'}$ and $B_{t',t,s}$. The final two components (Eqs. 7.3-7.4) are the non-anticipativity constraints. Non-anticipativity constraints for the first time period are written for all scenario pairs (r,s) such that, $r \neq s$. The NAC in Eq. 7.4 ensures that the decision variable $b_{t,s}$ is identical for scenarios (r,s) until the scenarios are differentiable. With endogenous uncertainty, two scenarios are differentiable when the indicator variable takes a value of 1. The value of the indicator variable, $Z_{t,s,s'}$, is given by Eq. 7.5. It is a function of $b_{i,t,s}$, the endogenous stage decision variables.

EN: The standard form of an MSSP with endogenous uncertainty.

$$\min \sum_{s \in \mathbb{S}} p_s \sum_{t \in \mathcal{T}} \left(\sum_{i \in \mathbb{I}} \left(D_{i,t} b_{i,t,s} \right) + f_{t,s} y_{t,s} \right) \tag{7.1}$$

$$\sum_{t' \in \mathcal{T}, t' < t} \left(\sum_{i \in \mathbb{I}} \left(G_{i,t,t',s} b_{i,t,s} \right) + B_{t',t,s} y_{t',s} \right) \leq a_{t,s} \quad \forall s \in \mathbb{S}, \forall t \in \mathcal{T} \tag{7.2}$$

$$b_{i,1,r} = b_{i,1,s} \quad \forall s, s' \in \mathbb{S}, \forall i \in \mathbb{I} \tag{7.3}$$

$$\left[Z_{t,r,s} \right] \vee \begin{bmatrix} \neg Z_{t,r,s} \\ y_{t,r} = y_{t,s} \\ b_{i,t+1,r} = b_{i,t+1,s} \end{bmatrix} \quad \forall r, s \in \mathbb{S}: r \neq s, \forall t \in T \tag{7.4}$$

$$Z_{t,r,s} \Leftrightarrow F\left(b_{i',1,s}, b_{i',2,s}, \ldots, b_{i',t,s} \right) \quad \forall r, s \in \mathbb{S}, \forall t \in T \tag{7.5}$$

$$Z_{t,r,s} \in \{0,1\} \qquad y_{t,s}, b_{i,t,s} \in \mathbb{R} \quad \forall t \in T, (r,s) \in \mathbb{S} \tag{7.6}$$

7.1.2 MSSPs with Endogenous and Exogenous Uncertainty

The extension of the MSSP framework to include both endogenous and exogenous uncertainty was originally presented in Goel and Grossmann (2006) .

Here, we include a brief description for completeness. From the endogenous problem, the planning horizon, $t \in \mathcal{T}$, the decision variables, $b_{i,t,s}$ and $y_{t,s}$, and the endogenous uncertain parameters, θ_i, persist. Let $\mathcal{J}$ be a set of exogenous uncertain parameters defined by $\mathcal{J} = [\mathcal{J}]$. We denote ξ_t as a vector of exogenous realizations defined for parameters $j \in \mathcal{J}$ and realized at time t. Each exogenous uncertain parameter is assumed to have a finite set of possible realization, $\mathcal{E}_{j,t} \in \left\{\xi_{j,t}^1, \xi_{j,t}^2, \dots, \xi_{j,t}^{\mathcal{R}}\right\}$. To handle exogenous parameters, we define a new variable $x_{t,s}$ which represents the exogenous stage decisions. Construction of the scenario set is performed similar to that of the endogenous problem, $\mathbb{S} \coloneqq \times_{t \in \mathcal{T}}\left\{\times_{j \in \mathcal{J}} \mathcal{E}_{j,t}\right\} \times \{\times_{i \in \mathbb{I}} \theta_i\}$ (Apap & Grossmann, 2016).

The objective function (Eq. 7.7) minimizes the cost of the endogenous stage decision variables and the recourse actions. Adding exogenous parameters introduces the cost of the exogenous stage decisions variable, $x_{t,s}$, in the objective function (Eq. 7). The scenario specific constraints (Eq. 7.8) are updated to reflect the addition of exogenous uncertainty. First stage NACs apply to both the endogenous and exogenous uncertainty, thus both the exogenous and endogenous variables must be identical at the first time period (Eqs. 7.9 and 7.10). Recalling that the realization of exogenous uncertain parameters is decision independent, i.e., the timing of the realizations are known *a priori*, the NACs for the exogenous parameters are written using a set of equality constraints shown in Eq. 7.11. The set $\boldsymbol{\tau}(r,s)$, representing the time period in which scenario r is distinguishable from scenario s, is generated such that for any scenario pair (r,s) the value $\boldsymbol{\tau}(r,s) = \max\left\{t \mid t' \in \mathcal{T} \text{ } and \text{ } \xi_{t'}^{s} = \xi_{t'}^{s'} \; \forall t' \leq t\right\}$. In the EN formulation if two scenarios r, s differ in the realization of an endogenous uncertain parameter a disjunction written. When the indicator variable $Z_{t,s,s'} = 0$ equality constraints are enforced to ensure that decision variables in scenarios r and s are identical. However, when the formulation is extended to incorporate both endogenous and exogenous uncertainty the disjunction changes. When two scenarios r and s differ in only endogenous uncertain parameters the disjunction is identical to the EN formulation. If scenarios r and s differ in both endogenous and exogenous parameters, the disjunction shown in Eq. 7.12 is used. The time periods for which the disjunction is written depend on the time period when scenarios r and s are differentiable by the exogenous uncertain parameter (given as $\boldsymbol{\tau}(r,s)$). The formulation for the MSSP with endogenous and exogenous uncertainty (ENEX) is shown below.

ENEX: The standard form of an MSSP with endogenous and exogenous uncertainty.

$$\min \sum_{s \in \mathbb{S}} p_s \sum_{t \in \mathcal{T}} \left(\sum_{i \in \mathbb{I}} \left(D_{i,t} b_{i,t,s} \right) + C_{t,s} x_{t,s} + f_{t,s} y_{t,s} \right) \tag{7.7}$$

$$\sum_{t' \in \mathcal{T}, t' < t} \left(\sum_{i \in \mathbb{I}} \left(G_{i,t,t's} b_{i,t,t'} \right) + A_{t',t,s} x_{t',s} + B_{t',t,s} y_{t',s} \right) \leq a_{t,s} \quad \forall s \in \mathbb{S}, \forall t \in \mathcal{T} \tag{7.8}$$

$$x_{1,r} = x_{1,s} \quad \forall r, s \in \mathbb{S} : r > s \tag{7.9}$$

$$b_{i,1,r} = b_{i,1,s} \quad \forall r, s \in \mathbb{S}, \forall i \in \mathbb{I} \tag{7.10}$$

$$\begin{bmatrix} x_{t+1,r} = x_{t+1,s} \\ y_{t,r} = y_{t,s} \\ b_{i,t+1,r} = b_{i,t+1,s} \end{bmatrix} \quad \forall r, s \in \mathbb{S} : r > s \quad \forall t \in \mathcal{T} : t \leq \boldsymbol{\tau}(r,s) \tag{7.11}$$

$$\left[Z_{t,r,s} \right] \vee \begin{bmatrix} \neg Z_{t,r,s} \\ x_{t+1,r} = x_{t+1,s} \\ y_{t,r} = y_{t,s} \\ b_{i,t+1,r} = b_{i,t+1,s} \end{bmatrix} \quad \forall r, s \in \mathbb{S} : r \neq s \wedge (r,s) \in \Psi, \forall t < \boldsymbol{\tau}(r,s)\ \boldsymbol{or}\ \mathcal{T}, i \in \mathbb{I} \tag{7.12}$$

$$Z_{t,r,s} \Leftrightarrow F\left(b_{i',1,s}, b_{i',2,s}, \ldots, b_{i',t,s} \right) \quad \forall r, s \in \mathbb{S}, \forall t \in \mathcal{T} \tag{7.13}$$

$$Z_{t,r,s} \in \{0,1\} \qquad y_{t,s}, b_{i,t,s} \in \mathbb{R} \quad \forall t \in \mathcal{T}, (r,s) \in \mathbb{S} \tag{7.14}$$

7.1.3 Linear Representation of Non-Anticipativity Constraints

Section 7.1.1 and 7.1.2 introduced the mathematical form of the MSSPs for problems with endogenous uncertainty (EN) and with endogenous and exogenous uncertainty (ENEX). To better illustrate the function of the endogenous NACs, specifically endogenous NACs with gradual realizations, we present a simple manufacturing example. Suppose that a company produces two different products, $p \in \{P1, P2\}$. Both products are required to complete three ordered processing stages using the same equipment. A product may develop a defect during any of the processing steps. If a product develops a defect, it does not complete the next step. Whether or not a product completes all processing stages without a defect is not known with certainty. We can represent this uncertainty associated with product p using an uncertain parameter θ_p. Thus, the possible outcomes for the uncertain parameter θ_p are $\Theta_p = \{\omega_p^1, \omega_p^2, \omega_p^3, \omega_p^4\}$ where ω_p^1, ω_p^2, and ω_p^3 represent a defect developed on product p in processing stages one, two, and three. The outcome ω_p^4 represents the case where the product p was successfully manufactured with no defects. In this example, the uncertainty is realized gradually, completing a processing stage will reveal if a defect is developed in that stage (a partial realization of uncertainty) but does not necessarily reveal whether or not the product will successfully complete all processing stages (complete realization of uncertainty). To have complete realization of uncertainty, all processing stages must be completed.

To illustrate how the non-anticipativity can be enforced for this example with the disjunctions given in the MSSP formulations (Eq 7.1 – Eq. 7.6), we begin by defining a binary variable, $\chi_{p,sg,t}^{s}$, which takes a value of 1 indicating that processing stage $sg \in [\boldsymbol{SG}]$ of product $p \in \boldsymbol{P}$ is started at time period $t \in \boldsymbol{T}$ for scenario $s \in \mathbb{S}$. Depending on the objective of the problem, the model may contain additional constraints or variables. These variables and constraints do not impact the development of NACs, as such, we omit them from this discussion. Prior to completing any processing stages all scenarios are indistinguishable, this implies that the values for the decision variable, $\chi_{p,sg,t}^{s}$, are identical until processing stages are completed.

Naturally it follows that we define a binary variable, $\zeta_{p,sg,t}^{s}$ which indicates whether the processing stage sg has been completed for product p at time t in scenario s. For simplicity in our example we assume that the each processing stage has a known completion time. We then can define Eq. 7.15, which relates the decision variable $\chi_{p,sg,t}^{s}$ to the indicator variable $\zeta_{p,sg,t}^{s}$ using the duration ($\delta_{p,sg}$) required to complete the processing stage sg for product p.

$$\zeta_{p,sg,t}^{s} = \zeta_{p,sg,t-1}^{s} + \chi_{p,sg,t-\delta_{p,sg}} \tag{7.15}$$

For every scenario pair, $\{r, s\} \in \mathbb{S} = \times_{p \in \boldsymbol{P}} \{\theta_p\}$, there exists one or more differentiating events. In the case of gradually realized uncertainty, the differentiating event is an event which causes a partial realization of an uncertain parameter. We demonstrate this concept using the manufacturing example. Consider two scenarios with the uncertain parameter realizations $(\omega_{P1}^{1}, \omega_{P2}^{3})$ and $(\omega_{P1}^{2}, \omega_{P2}^{4})$. The first scenario represents the case where product $P1$ develops a defect after processing stage 1 and product $P2$ develops a defect after processing stage 3. In the second scenario, product $P1$ develops a defect after processing stage 2 and product $P2$ successfully completes all processing stages without developing a defect. Before the products complete any processing stages the scenarios are not differentiable. Once product $P1$ completes first processing stage, these two scenarios become differentiable. If product $P1$ develops a defect during processing stage 1, the underlying value for ω_{P1} is realized to be ω_{P1}^{1}. If no defect is developed the value of ω_{P1} is realized not to be ω_{P1}^{1}. To realize the value of ω_{P1}, $P1$ must complete more processing stages .

For any two scenarios (r, s), we can establish that at each time period two scenarios are indistinguishable if no differentiating events have occurred previously. Whether or not a differentiating event has occurred can be described using Eq. 3.20. The set $\Psi(r, s)$ represents the set of differentiating events that can occur before scenarios r and s are differentiable. In the case of the manufacturing example, these events correspond to the completion of a particular processing stage. Equation 7.16 corresponds to Eq. 7.5 in EN.

$$Z_{t,r,s} = \sum_{(p',sg')\in\Psi(r,s)} \zeta_{p',sg',s} \tag{7.16}$$

In Eq. 7.16, the variable $Z_{t,r,s}$ takes a value of zero if the scenarios (r,s) are not differentiable and a positive integer otherwise. Equation 7.17 uses the value of $Z_{t,r,s}$ established in Eq. 3.20 to construct a disjunction. The disjunction enforces equality constraints on the problem, and ensures that the decision variables in scenarios r and s are identical (i.e. $Z_{t,r,s} = 0$). The disjunction presented in Eq. 7.17 functions as the disjunction shown in Eq. 7.4 in EN.

$$\left[Z_{t,r,s}\right] \vee \begin{bmatrix} \neg Z_{t,r,s} \\ \chi^{s}_{p,sg,t} = \chi^{r}_{p,sg,t} \end{bmatrix} \quad \forall r,s \in \mathbb{S}: r \neq s\,, \forall t \in \boldsymbol{T} \tag{7.17}$$

The disjunction can be converted to a linear constraint using 'big-M' representation. From Eq. 7.18, it can be seen that when $Z_{t,r,s}$ is equal to zero, the decision variables are identical in scenarios s and r.

$$\left|\chi^{s}_{p,sg,t} - \chi^{r}_{p,sg,t}\right| \leq M \cdot Z_{t,r,s} \quad \forall\, p \in \boldsymbol{P},\ sg \in \boldsymbol{SG},\ t \in \boldsymbol{T},\ (r,s) \in \mathbb{S} : r \neq s \tag{7.18}$$

When $Z_{t,r,s}$ is no longer equal to zero, the decision variables can take different values. It follows that the value of M is calculated as $\mathbf{max}\,|\chi^{s}_{p,sg,t} - \chi^{r}_{p,sg,t}|$. In addition to conditional non-anticipativity constraints, MSSPs with endogenous uncertainty also contain a NAC which ensures decision variables are identical at the first time period (Eq. 7.3 in EN). This constraint can be seen in Eq. 7.19.

$$\chi^{s}_{p,sg,1} = \chi^{r}_{p,sg,1} \quad \forall p \in \boldsymbol{P}, sg \in \boldsymbol{SG}, r \in \mathbb{S}: r \neq s \tag{7.19}$$

Goel and Grossmann (2006) established that if the scenario set is given as the Cartesian product, the constraints written in Eq. 7.20, only need to be written for any two scenarios (r,s) and not for scenarios (s,r). This property referred to as the symmetry property holds because the indicator variable $Z_{t,r,s}$ is identical to $Z_{t,s,r}$ (Eq. 7.20)

$$Z_{t,r,s} = Z_{t,s,r}. \tag{7.20}$$

Boland et al. (2016) showed this property to be applicable also for the sets of scenarios which are subsets of the Cartesian product (Boland et al., 2016).

An analogous framework can be presented to construct linear representation for the situations with both endogenous and exogenous uncertainty based on formulation

7.2 Proposed Approach for Generation of NACs for Scenario Subsets

7.2.1 Graph-based model for NACs

Generation of NACs relies on knowledge of the scenario structure and how the groupings of indistinguishable scenarios change with the realization of uncertainty. The goal of this work is to find an efficient approach to generate a minimum cardinality set of NACs, which we will denote by $\mathcal{N}$. We begin by projecting the scenarios onto a undirected lattice graph $\mathcal{G} = (\mathcal{S}, E)$. Here we assume that vertices represent scenarios $s \in \mathcal{S} \subseteq \mathbb{S}$ and the edges will represent the conditional NACs which relate pairs of scenarios. Before discussing the algorithm to generate the minimum cardinality set $\mathcal{N}$, we will establish some properties of the graph and how the grouping of vertices on the graph change as the knowledge of the underlying values of the uncertain parameters in the system changes.

For each uncertain parameter, $\in \mathbb{I} \cup \mathbb{J}$, there exist a set of events, $\mathcal{E}_\sigma$, which represent the events that must occur to fully realize the uncertain parameter σ. In the case of instantaneous uncertainty realization $|\mathcal{E}_\sigma| = 1$ indicating that with a single event the underlying value of the uncertain parameter is realized. When uncertainty is realized gradually, the definition of $\mathcal{E}_\sigma$ is less straightforward. If the gradual realization of uncertainty requires the set of events to be completed in a specific order similar to the manufacturing example, $\mathcal{E}_\sigma$ is defined as an ordered set, $\mathcal{E}_\sigma := \{e_\sigma^1, \dots, e_\sigma^{K_\sigma}\}$.

Definition 1

Let ρ *be the power set of all events, i.e.,* $\rho = 2^{\cup_{\sigma \in \mathbb{I}\cup\mathbb{J}} \mathcal{E}_\sigma}$. *We will call a* $\mathrm{c} \in \rho$ *a cut. When applied to graph* $\mathcal{G}$ *each cut generates a set* $\mathfrak{G}_\mathrm{c}$ *of mutually disjoint subsets of vertices* $\mathfrak{s} \in \mathfrak{G}_\mathrm{c}$, $\bigcup \mathfrak{s} = \mathcal{S}$, *such that scenarios within each* $\mathfrak{s}$ *are indistinguishable given* c. *Further, we will denote by* $\mathcal{C} \subset \rho$ *the set of permissible cuts.*

The primary benefit of the lattice graph is that the cuts applied to the graph are either vertical or horizontal by construction. Illustrating this concept with the manufacturing example, we begin by constructing the lattice. Here we show the complete scenario set, however the concept can be generalized to any subset of the full scenario set. Figure 7.1a shows the scenarios projected onto the lattice structure for the manufacturing example. Recall that for each product there are four possible outcomes, resulting in a total of 16 scenarios.

If there is no order to the gradual realizations (or the realizations are instantaneous), the cut set formed is the power set of $\mathcal{E}_\sigma$, i.e. $\mathcal{C} = \rho$. When there is an of order associated with the gradual realization of uncertainty, i.e., $\mathcal{E}_\sigma$ is an ordered set, the set of cuts $\mathcal{C}$ forms a proper subset of ρ as not all sets of events are permissible. Clearly, in this case, for any set of cuts $\mathrm{c} \in \mathcal{C}$, if an event $\mathrm{e} \in \mathrm{c}$, then all events e' that should have occurred prior to e are also in c. The sets of scenarios $\mathfrak{s} \in \mathfrak{G}_\mathrm{c}$ formed by each cut $\mathrm{c} \in \mathcal{C}$ are generated by dividing the full scenario set. For all scnearios in that subset, the scenarios share identical information for all events. To illustrate this concept, we use two cut sets from the manufacturing example. The first cut set $\{\mathrm{e}_{\mathrm{P}1}^{1}\}$, divides the scenario set into two groups based on event 1 for product *P1*. In the first group, all scenarios develop a defect during processing stage 1. In the second group all scenarios do not develop a defect during processing stage 1. The division of scenarios is shown in Fig. 7.1b. Figure 7.1c shows a second example of scenario division. The cut set for Fig. 7.1c contains events $\mathrm{e}_{\mathrm{P}1}^{1}$ and $\mathrm{e}_{\mathrm{P}2}^{2}$. The two events divide the scenario set into four groups. One set contains a single scenario and represents the case where both products develop a defect in the first processing stage. Two scenario sets contain three scenarios each which represent the cases where one product develops a defect in processing stage one. The final group contains nine scenarios where defects do not develop in the first processing stage.

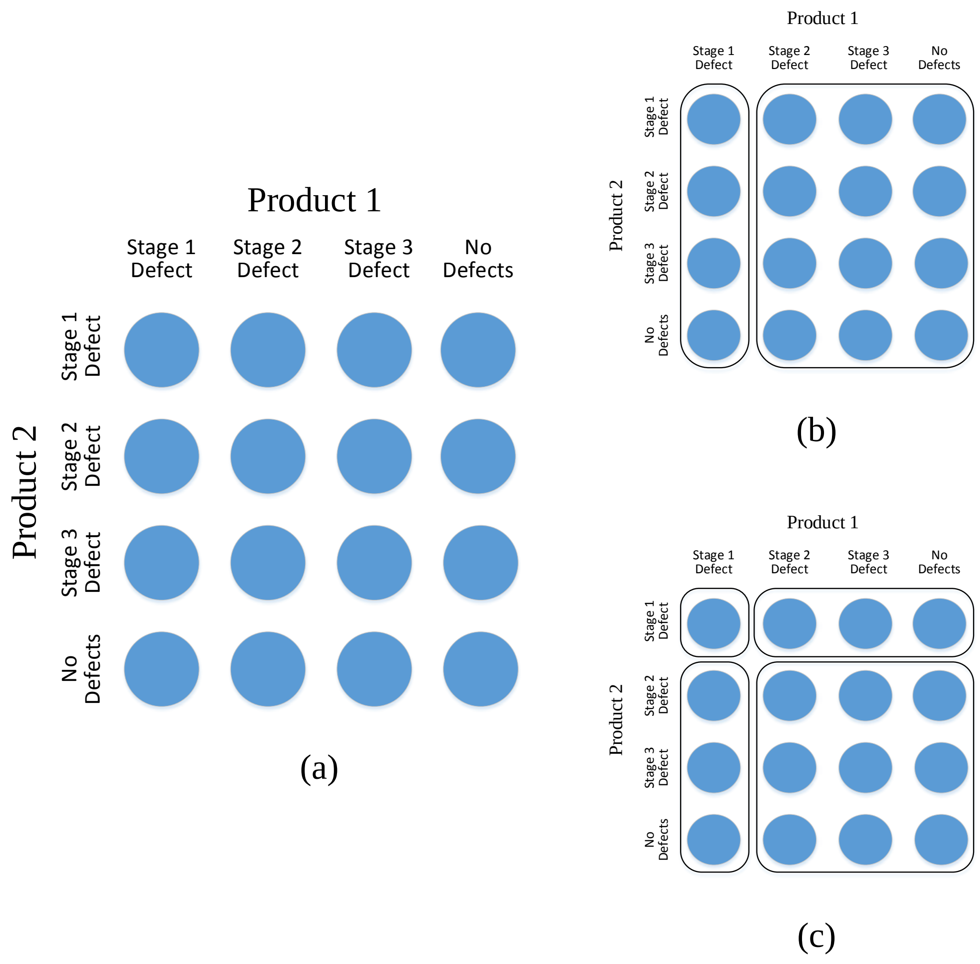


Figure 7.1 (A) Projection of the full set of scenarios in the manufacturing example onto a lattice graph (B) Scenario groups of the manufacturing example formed by cut set $\{e_{P1}^{1}\}$. (C) Scenario groups formed by the cut set $\{e_{P1}^{1}, e_{P2}^{1}\}$

Remark

The division of the scenario set into subsets based on the cut sets differ slightly if the realization of uncertainty is instantaneous. In the manufacturing example, an event would add a single cut to the graph $\mathcal{G}$. In contrast, instantaneous realization of a parameter θ_i would results in $|\Theta_i|$ cuts added to the graph $\mathcal{G}$. The cuts would create groups based on the realized value of the parameter θ_i.

Definition 2

We will say that a subset of scenarios $\mathfrak{s}$ is connected if the corresponding restriction of graph $\mathcal{G}$ is connected.

Next, we will describe the relationship between a set of constraints sufficient to enforce non-anticipativity and connectivity of subsets of nodes. At the initial time period no uncertainty no uncertainty has been realized. This implies that all scenarios are identical, thus there is one group of scenarios (the whole set). Throughout the planning horizon uncertainty is realized. As uncertainty is realized, the scenario set is divided into subsets based on the realization of uncertainty. Figure 7.2 shows the progression of uncertainty realization in the manufacturing example. Below each vertex set presented in Fig. 7.2, the cut set $c \in \mathcal{C}$ corresponding to the vertex set is shown.

Definition 3

We will say the degree of uncertainty realization of any cut set $\mathrm{c} \in \mathcal{C}$ *is given by* $|\mathrm{c}|$*, or in other words,* $|\mathrm{c}|$ *represents the number of differentiating events which have occurred in order to reach the state of uncertainty realization.*

Proposition 1

Any set of subsets $\mathfrak{S}_{\mathrm{c}}: \mathfrak{s} \in \mathfrak{S}_{\mathrm{c}}$ with order $|\mathrm{c}|$ can be transformed into a set of subset $\mathfrak{S}_{\mathrm{c}'}: \mathfrak{s}' \in \mathfrak{S}_{\mathrm{c}'}$ with order $|\mathrm{c}| + 1$ if $|\mathrm{c}' \backslash \mathrm{c}| = 1$. Moreover, $\mathfrak{s}' \subset \mathfrak{s}$ if $\mathfrak{s} \cap \mathfrak{s}' \neq \emptyset$ for all $\mathfrak{s} \in \mathfrak{G}_{\mathrm{c}}$ and $\mathfrak{s}' \in \mathfrak{G}_{\mathrm{c}}$.

Proof

The order of the cut set refers to the number of differentiating events which have occurred to reach the state of uncertainty realization. Two scenarios $r, s \in \mathfrak{s} \in \mathfrak{G}_c$ if they are indistinguishable with respect to the differentiating events in c. Increasing the number of differentiating events in the cut set c (i.e. $|c'| = |c| + 1$ and $c \subset c'$) results in formation of a new set of subsets $\mathfrak{G}_{c'}$. If two scenarios $\{s, s'\} \notin \mathfrak{s} \in \mathfrak{S}_c$ then the event differenting scenarios s and s' exists in the set c. This implies that if an additional differentiating event is added to the cut set c to form a cut set c', then $\{s, s'\} \notin \mathfrak{s} \in \mathfrak{S}_{c'}$.

If two scenarios $\mathrm{r}, \mathrm{s} \in \mathfrak{s} \in \mathfrak{G}_\mathrm{c}$ for some cut $\mathrm{c} \in \mathcal{C}$, then they are indistinguishable for some permissible state of the system, and hence, non-anticipativity should be enforced for these two scenarios. On the other hand, a direct non-anticipaticity constraint for (r, s) can be eliminated if another path $\mathcal{P}$ exists within $\mathfrak{s}$ connecting r and s due to transitivity of the constraints. Further, since while uncertainty is gradually realized new scenario sets are constructed as subsets of existing sets, the sets formed by the differentiating event must exist as connected sets prior to the occurrence of the differentiating event. This observation leads to the following characterization of relationship between NACs and lattice graphs.

Proposition 2

A set of constraints is sufficient to enforce non-anticipativity, if and only if for all cuts $\mathrm{c} \in \mathcal{C}$ *all of the corresponding scenario subsets* $\mathfrak{s} \in \mathfrak{G}_\mathrm{c}$ *are connected.*

Proof

Follows by construction, from the discussion above.

Next, we will demonstrate how using the knowledge of the structure of the subsets of vertices formed by each cut set and the relationship between cut sets, it is possible to develop an algorithm which generates a set of edges for the graph $\mathcal{G}$ such that it satisfies Proposition 2 with the minimal number of corresponding NACs.

Definition 4

An edge $(\mathrm{r}, \mathrm{s}) \in \mathrm{E}$ *will be called necessary for a set of nodes* $\mathfrak{s} \in \mathfrak{G}_\mathrm{c}$ *if* $\mathrm{r}, \mathrm{s} \in \mathfrak{s}$, *and* $\mathrm{r}, \mathrm{s} \notin \mathfrak{s}'$ *for all* $\mathfrak{s}' \in \mathfrak{G}_{\mathrm{c}'}$ *and* $\mathrm{c}' \in \mathcal{C}$ *such that* $|\mathrm{c}| < |\mathrm{c}'|$*:*

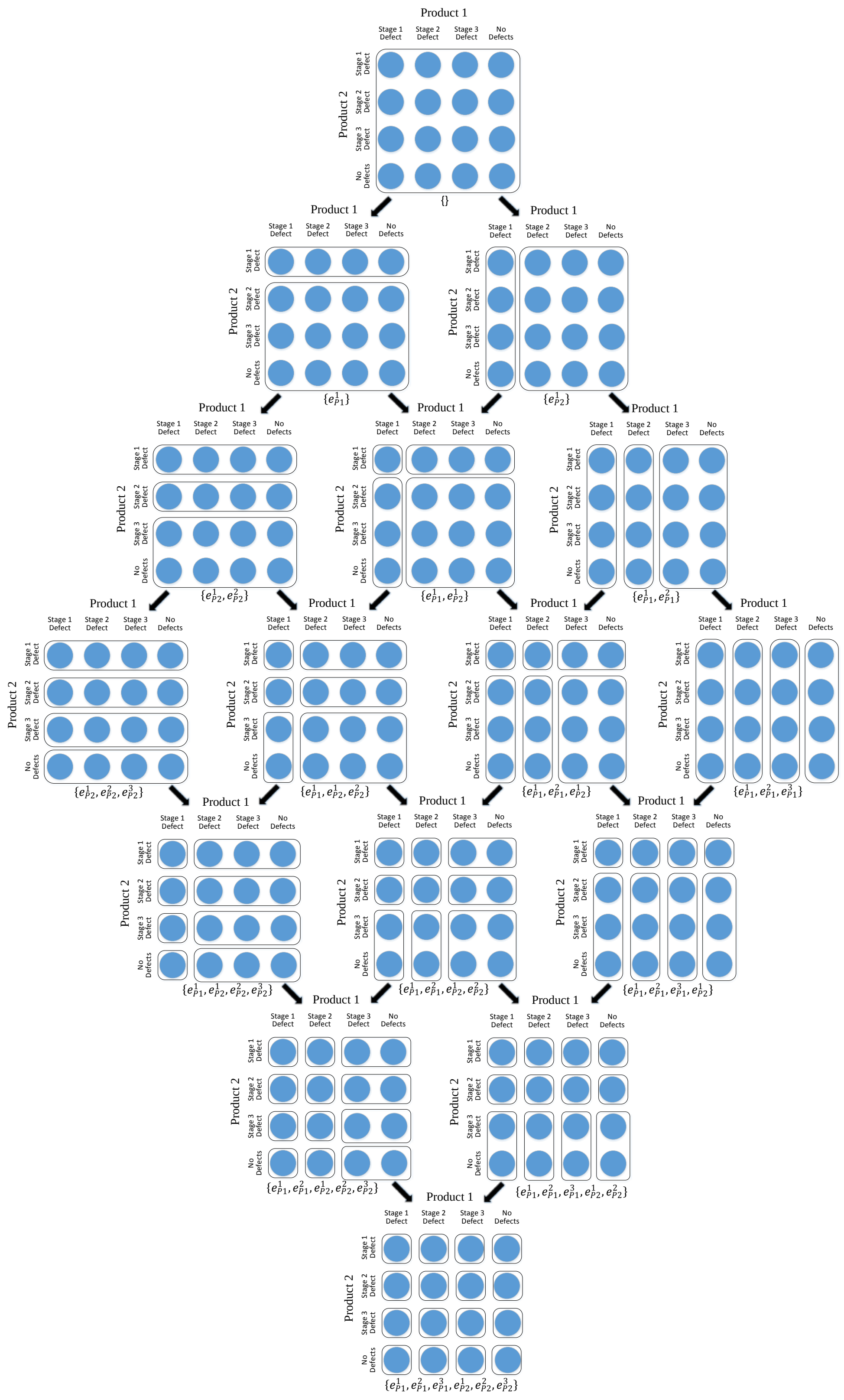

Product 1
Stage 1 Defect
Stage 2 Defect
Stage 3 Defect
No Defects
Product 2
{}
{e^1_{P1}}
{e^1_{P2}}
{e^1_{P2}, e^2_{P2}}
{e^1_{P1}, e^1_{P2}}
{e^1_{P1}, e^2_{P1}}
{$e^1_{P2}, e^2_{P2}, e^3_{P2}$}
{$e^1_{P1}, e^1_{P2}, e^2_{P2}$}
{$e^1_{P1}, e^2_{P1}, e^1_{P2}$}
{$e^1_{P1}, e^2_{P1}, e^3_{P1}$}
{$e^1_{P1}, e^1_{P2}, e^2_{P2}, e^3_{P2}$}
{$e^1_{P1}, e^2_{P1}, e^1_{P2}, e^2_{P2}$}
{$e^1_{P1}, e^2_{P1}, e^3_{P1}, e^1_{P2}$}
{$e^1_{P1}, e^2_{P1}, e^1_{P2}, e^2_{P2}, e^3_{P2}$}
{$e^1_{P1}, e^2_{P1}, e^3_{P1}, e^1_{P2}, e^2_{P2}$}
{$e^1_{P1}, e^2_{P1}, e^3_{P1}, e^1_{P2}, e^2_{P2}, e^3_{P2}$}

Figure 7.2 The sets formed by each cut set $c \in \mathcal{C}$ for the manufacturing example

By this definition an edge (r, s) is necessary for $\mathfrak{s} \in \mathfrak{G}_c$ if non-anticipativity for scenarios r, s is required (as $r, s \in \mathfrak{s}$), yet it cannot be established based on edges that are within subsets $s' \in \mathfrak{G}_{c'}$ for $|c'| > |c|$, i.e., when more information on uncertainty realization is available. In other words, unless non-anticipativity for r, s is enforced with a path within $\mathfrak{s}$, then it will not follow for any other set of NACs. Note that it is not required in the definition above to $|c' \backslash c| = 1$, as non-anticipativity can be enforced based on paths in $\mathcal{G}$ created based on a cut set which has a higher order in lieu of adding a new edge.

Definition 5

Any set of necessary edges E' *required to ensure the connectivity of* $\mathfrak{s}$ *is said to be minimal if there exists no other set of edges* E'' *which also guarantees connectivity in* $\mathfrak{s}$*, such that* $\sum_{(\mathrm{r,s}) \in \mathrm{E}''} |\Psi(\mathrm{r,s})| + |\tau(\mathrm{r,s})| < \sum_{(\mathrm{r,s}) \in \mathrm{E}'} |\Psi(\mathrm{r,s})| + |\tau(\mathrm{r,s})|$.

Note that the value of $|\Psi(\mathrm{r,s})| + |\tau(\mathrm{r,s})|$ represents the number of NACs required for a pair of scenarios $\mathrm{r,s}$.

Proposition 3

Any graph $\mathcal{G}$ *such that all subsets of nodes* $\mathfrak{s} \in \mathfrak{G}_\mathrm{c}$ *are connected and the restriction of* $\mathcal{G}$ *on any of these subsets of nodes is composed of only minimal necessary edges provides a minimum cardinality NAC set.*

Proof

The result follows immediately from the definitions of necessary edges and minimal necessary edge sets. Suppose there is another graph $\mathcal{G}'$ that does not consist of only minimal, yet provides a lower cardinality set of NACs than $\mathcal{G}$. Clearly, all edges in such a graph have to be necessary (in the sense of Definition 4), as any non-necessary can be removed without any change in the validity of NACs, which would reduce the cardinality of the NAC set. Further, then we can substitute all edges in this restriction with minimal edges. This would have no effect on non-anticipativity for any other subsets $\mathfrak{s}'$ since $\mathfrak{s}$ will remain connected, and hence the new graph will imply a reduction in the total cardinality.

Note that this result does not describe how such a graph can be constructed. In the next Section we will present an algorithm for constructing a graph $\mathcal{G}$ which satisfies the requirements in Proposition 3, at the same time proving its existence.

7.2.2 Sample non-anticipaticity constraint (SNAC) algorithm to find minimum cardinality NAC set

The proposed algorithm is presented in Algorithm 1. It is constructed as a greedy procedure, which begins by considering fully realized uncertainty sets ($k = \left|\bigcup_{\sigma\in\mathbb{I}\cup\mathbb{J}} \mathcal{E}_\sigma\right|$) and then generates minimal subsets of necessary edges separately for each possible cut set $c \in \mathcal{C}$ such that $|c| = k$. The process of generating subsets is repeated for cut set with $|c| = k$. Any edge deemed necessary is stored, then the value of k is decreased. The process is repeated until $k = 0$. The minimum cardinality NAC set is represented by edges deemed necessary in all iterations. For each $\mathfrak{s}$ the set of minimal necessary edges is found as a Minimum Spanning Tree (MST), taking into account any of the previously identified necessary edges. The weight of the edges correspond to $|\Psi(r,s)| + |\tau(r,s)|$, i.e., the number of NACs for a pair of scenarios r, s. An example progression of the algorithm is presented in section 7.2.3.

Theorem 1

Algorithm 1 terminates with a subset of edges which correspond to a minimum-cardinality set of NACs. The algorithm can be implemented in a way that guarantees that the time complexity grows as $O(|\mathcal{C}|)O(\mathcal{S}^3)$ as $|\mathcal{C}|, \mathcal{S} \to +\infty$, where $\mathcal{S}$ is the number of the scenarios and $\mathcal{C}$ is the set of permissible cuts.

Proof

The correctness result follows by construction from Proposition 3. Observe that since the algorithm proceeds from the state of full uncertainty, edges added earlier can be used in lieu of new edges. Hence, as long as previously added edges are taken into account, MST procedure on subset $\mathfrak{s}$ by definition outputs a minimal set of necessary edges for $\mathfrak{s}$, i.e., a subset of edges that enforce connectivity at minimum cost which implies the result.

In order to establish time-complexity, observe that the algorithm iterate $c \in \mathcal{C}$ and then over all $\mathfrak{s} \in \mathfrak{G}_c$. For each $\mathfrak{s}$ it then generates the set of scenarios in $\mathfrak{s}$ and performs an MST procedure. The number of subsets $\mathfrak{s}$ for each c is bounded by S. In a simplest implementation an MST can be found in $O(n^2)$, where n is the number of nodes in a graph. In our case, the number of nodes in each subset $\mathfrak{s}$ is bounded by S. Further, generation of each subset $\mathfrak{s}$ can be organized in $O(S)$ by enumeration. This then results in overall $O(|C|S^3)$ complexity.

Remark

Note that MST can be identified faster than $O(n^2)$, depending on the implementation used. Further, for a lattice graph even more efficient implementations are possible.

Remark

Each component $\mathfrak{s} \in \mathfrak{G}_c$ for $|c| = k$ is treated “in parallel”, i.e., MSTs are constructed without regard to edges that are added at the same iteration of k. This can lead to creation of cycles in $\mathcal{G}$. When MST procedure is applied to any subset, each connected component is treated as a single node.

The running time is proportional to $O(|\mathcal{C}|)$ which in the worst case is a powerset of the set of uncertainty realization events, i.e., it is exponential in the number of events. On the other hand, if the number of uncertain parameters and events is fixed, and $S \ll |\mathcal{C}|$, which is often the case, then the running time is polynomial in the number of scenarios.

```
$k := \left| \bigcup_{\sigma \in I \cup J} E_\sigma \right|$
$\mathcal{N} := \emptyset$
$while\ k \geq 0$
        for $c \in \mathcal{C} : |c| = k$
            STEP 1: Generate subsets $\mathfrak{s} \in \mathfrak{S}_c$
            STEP 2: Use MST to identify necessary edges for each subset $\mathfrak{s}$ with
                    existing edges $\mathcal{N}$
            STEP 3: Add edges identified in STEP 2 to $\mathcal{N}$
        $k = k - 1$
```

Algorithm 1 The Sample Non-Anticipativity Constraint (SNAC) Algorithm for minimum cardinality NAC generation

7.2.3 Illustration of the Algorithm Progression for a Simple Manufacturing Example

The manufacturing problem considered is described in detail in Section 7.1.3. The cardinality of the full set of scenarios for a manufacturing problem with two products and three processing stages is 16 (calculated as $4^2 = 16$). For this example, we sample 6 random scenarios, $\mathcal{S} \in \{(\omega_{P1}^1, \omega_{P2}^1), (\omega_{P1}^4, \omega_{P2}^3), (\omega_{P1}^2, \omega_{P2}^1), (\omega_{P1}^3, \omega_{P2}^2), (\omega_{P1}^4, \omega_{P2}^1), (\omega_{P1}^3, \omega_{P2}^3)\}$. Projection of these scenarios can be seen in Fig. 7.3.

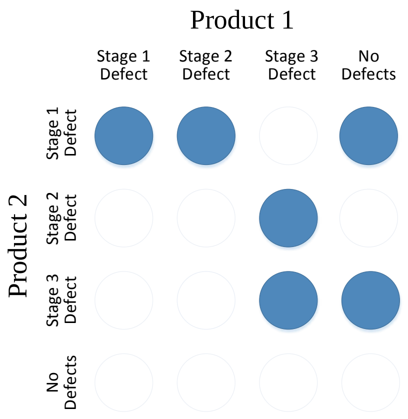


Figure 7.3 The projection of the six sampled scenarios onto the integer lattice

Since the uncertainty is realized gradually and there is an order to the realizations, the set $\mathcal{C}$ is a proper subset of the power set of the full set of differentiating events. The set of differentiating events is calculated as $\cup_{\sigma \in \{\omega_{P1}, \omega_{P2}\}} \mathcal{E}_\sigma$, where $\mathcal{E}_\sigma \coloneqq \{e_p^1, e_p^2, e_p^3\}$ where e_p^1 represents the completion of processing stage 1 for product p. Table 7.1 shows the cut set $\mathcal{C}$.

Table 7.1 The cut sets $\mathcal{C}$ for the manufacturing example

Order	$\mathcal{C}$
0	{}
1	$\{e_{P1}^1\}$, $\{e_{P2}^1\}$
2	$\{e_{P2}^1, e_{P2}^2\}$,$\{e_{P1}^1, e_{P2}^1\}$, $\{e_{P1}^1, e_{P1}^2\}$
3	$\{e_{P2}^1, e_{P2}^2, e_{P2}^3\}$, $\{e_{P1}^1, e_{P2}^1, e_{P2}^2\}$, $\{e_{P1}^1, e_{P1}^2, e_{P2}^1\}$, $\{e_{P1}^1, e_{P1}^2, e_{P1}^3\}$
4	$\{e_{P1}^1, e_{P1}^2, e_{P2}^1, e_{P2}^2\}$,$\{e_{P1}^1, e_{P1}^2, e_{P1}^3, e_{P2}^1\}$ $\{e_{P1}^1, e_{P2}^1, e_{P2}^2, e_{P2}^3\}$
5	$\{e_{P1}^1, e_{P1}^2, e_{P2}^1, e_{P2}^2, e_{P2}^3\}$,$\{e_{P1}^1, e_{P1}^2, e_{P1}^3, e_{P2}^1, e_{P2}^2\}$
6	$\{e_{P1}^1, e_{P1}^2, e_{P1}^3, e_{P2}^1, e_{P2}^2, e_{P2}^3\}$

The algorithm starts by initializing k to a value of $| \cup_{\sigma \in \{\omega_{P1}, \omega_{P2}\}} \mathcal{E}_\sigma |$. In this case, the value of k is set to six. The next step is to divide the scenarios into subsets for each $c \in \mathcal{C}: |c| = k$. The cut set c with $|c|$=6 produces subsets where each scenario forms its own subset. As a result, no edges are deemed necessary. Reducing k by one and generating new subsets results in the subsets shown in Fig. 7.4. The corresponding cut set with five elements is shown below each set of subsets. In Fig. 7.4(A), two scenarios fall into the same subset and MST provides the list of necessary edges represented with a dotted line.

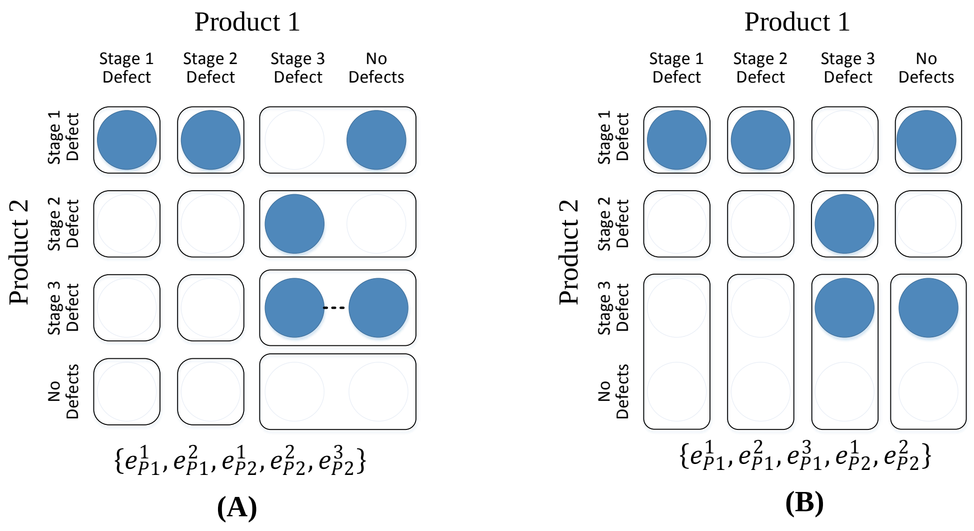


Figure 7.4 Subsets formed by the cut sets of length five

After identifying the necessary edges, the edges are added to the set $\mathcal{N}$, and the value of k is decreased by one. Subsets are then generated for cut sets with four elements. These sets are shown in Fig. 7.5. The edges which were deemed necessary in the previous iteration are shown as solid lines in Fig. 7.5. In Fig. 7.5(B), the edges adding during the previous iterations are sufficient to satisfy property 1 for all subsets. In both Fig. 7.5(A) and 7.5(C), there exists at least one subset where additional edges are needed. The additional edges deemed necessary by the MST for each subset are shown with dotted lines. The edges deemed necessary are then added to the set $\mathcal{N}$.

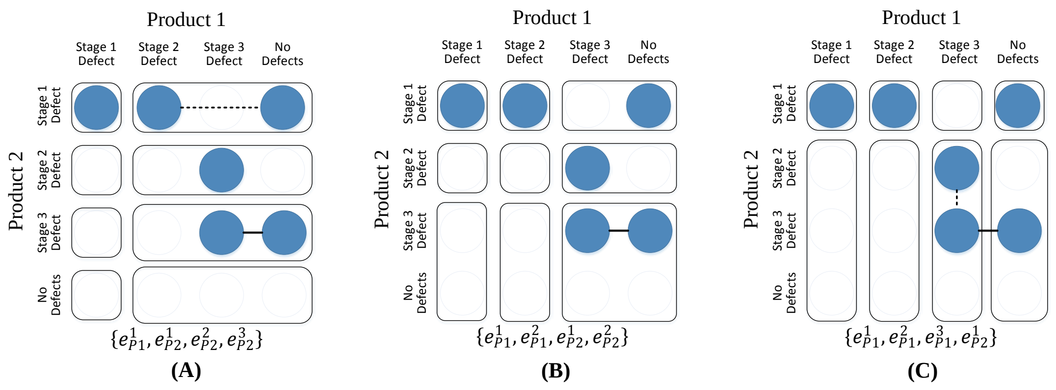

Product 1
Stage 1 Defect
Stage 2 Defect
Stage 3 Defect
No Defects
Product 2
Stage 1 Defect
Stage 2 Defect
Stage 3 Defect
No Defects
$\{e_{P1}^1, e_{P2}^1, e_{P2}^2, e_{P2}^3\}$
(A)
$\{e_{P1}^1, e_{P1}^2, e_{P2}^1, e_{P2}^2\}$
(B)
$\{e_{P1}^1, e_{P1}^2, e_{P1}^3, e_{P2}^1\}$
(C)

Figure 7.5 Subsets for the iteration where $k = 4$

The value of k is decreased again, and iterations continue until the value of k is equal to zero. The subsets and necessary edges for the remaining iterations are summarized in Fig. 7.5. The iteration with $k = 3$ adds the final two edges to the necessary set of edges. The algorithm generated a total of five non-anticipativity constraints to connect six scenarios. Without reductions the number of NACs would have been 15 calculated by $0.5 \cdot [|\mathcal{S}| \cdot (|\mathcal{S}| - 1)]$.

7.3 Computational Studies

In this section, we present results of a computational study investing the SNAC algorithm run time and the number of NACs generated by the algorithm. he study compares the number of NACs generated by SNAC algorithm to the number of NACs that will be added to the MSSP without any reductions. All computational studies were implemented in Python Version 3.5.0 and performed on a 64-bit machine running Windows 7 with Xeon E3-1241 with 32 GB RAM.

We vary the number of uncertain parameters, the number of outcomes that each parameter can take, and the number of scenarios considered. In each case, all uncertain parameters are endogenous with gradual realizations of uncertainty where the realization of uncertainty is ordered. The scenarios are then randomly sampled from the Cartesian product. For each set of parameter values 30 random instances were generated. Figure 7.7 plots the average running time of the SNAC algorithm for each set of parameter values.

In Fig. 7.7, as the number of uncertain parameters increases so does the algorithm running time. Considering the time complexity of the algorithm, this is expected. Additional uncertain parameters increase the number of cut sets which need to be considered when determining the minimum cardinality set. Similarly, the algorithm running time also increases when the number of scenarios is increased. Note that for larger values of the number of scenarios (larger than 100) we observe a linear growth of running time in log-log coordinates which corresponds to polynomial time complexity.

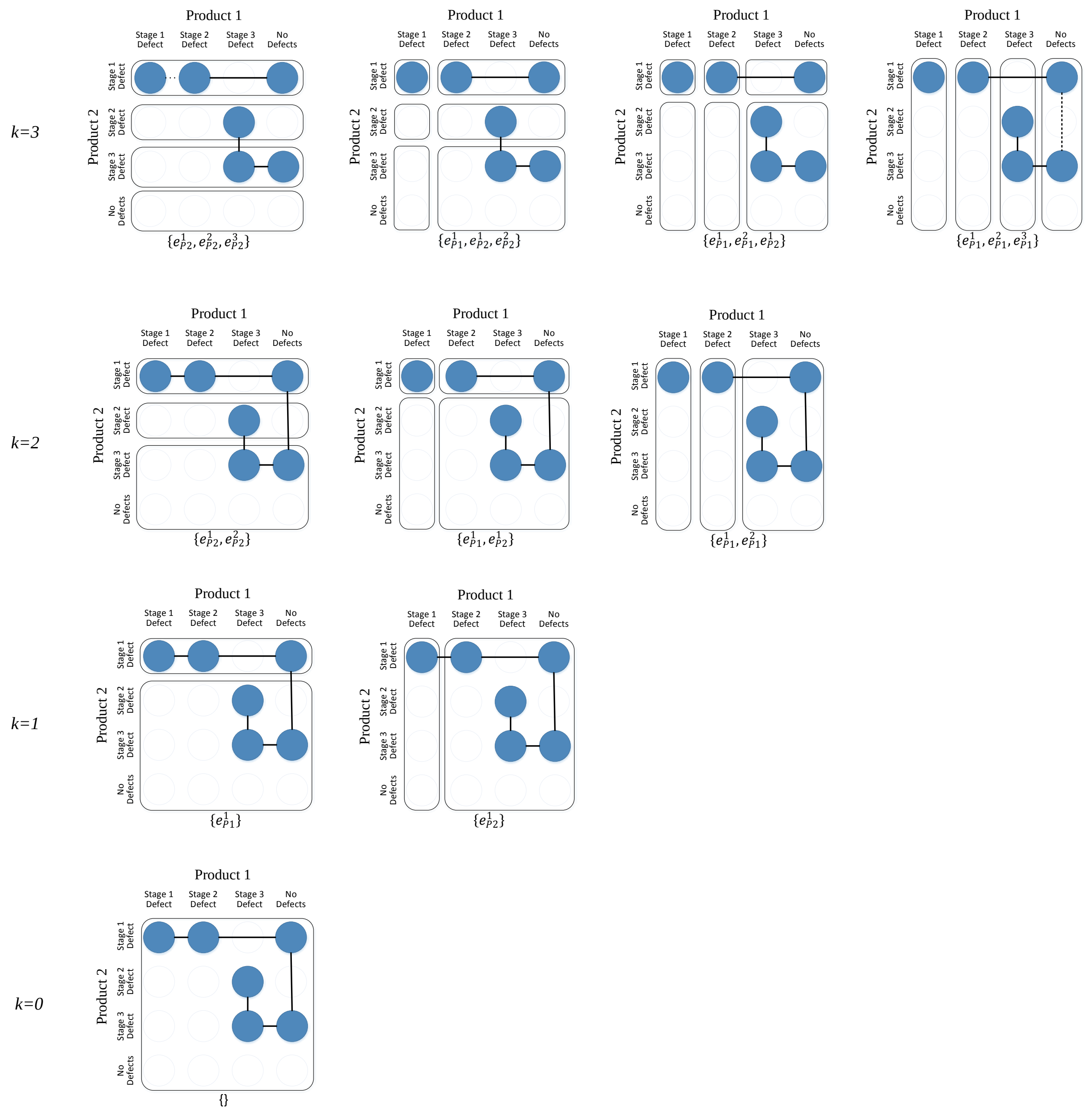


Figure 7.6 The last four iterations of the SNAC algorithm for the two product manufacturing example

Non-anticipativity constraint reduction using the SNAC algorithm is dependent on the number of scenarios. In Table 7.2, the number of NACs that are generated without the SNAC algorithm (NACs without Reductions) represent the number of different handshakes for the hand shake problem with $|\mathcal{S}|$ people calculated using

$0.5[|\mathcal{S}|(|\mathcal{S}|-1)]$. The SNAC algorithm uses knowledge of the relationships between scenarios to make substantial reductions in the number of NACs required. Table 7.2 illustrates the reduction observed for several case studies. For instance, with six scenarios there are 15 NACs without reduction. The SNAC algorithm returns a maximum of nine NACs resulting in a 40% reduction. Contrasting that to the case where there are 512 scenarios, the number of NACs without reduction total 130,816. The SNAC algorithm finds that at most 3338 NACs are needed. The result is more than a 97% reduction.

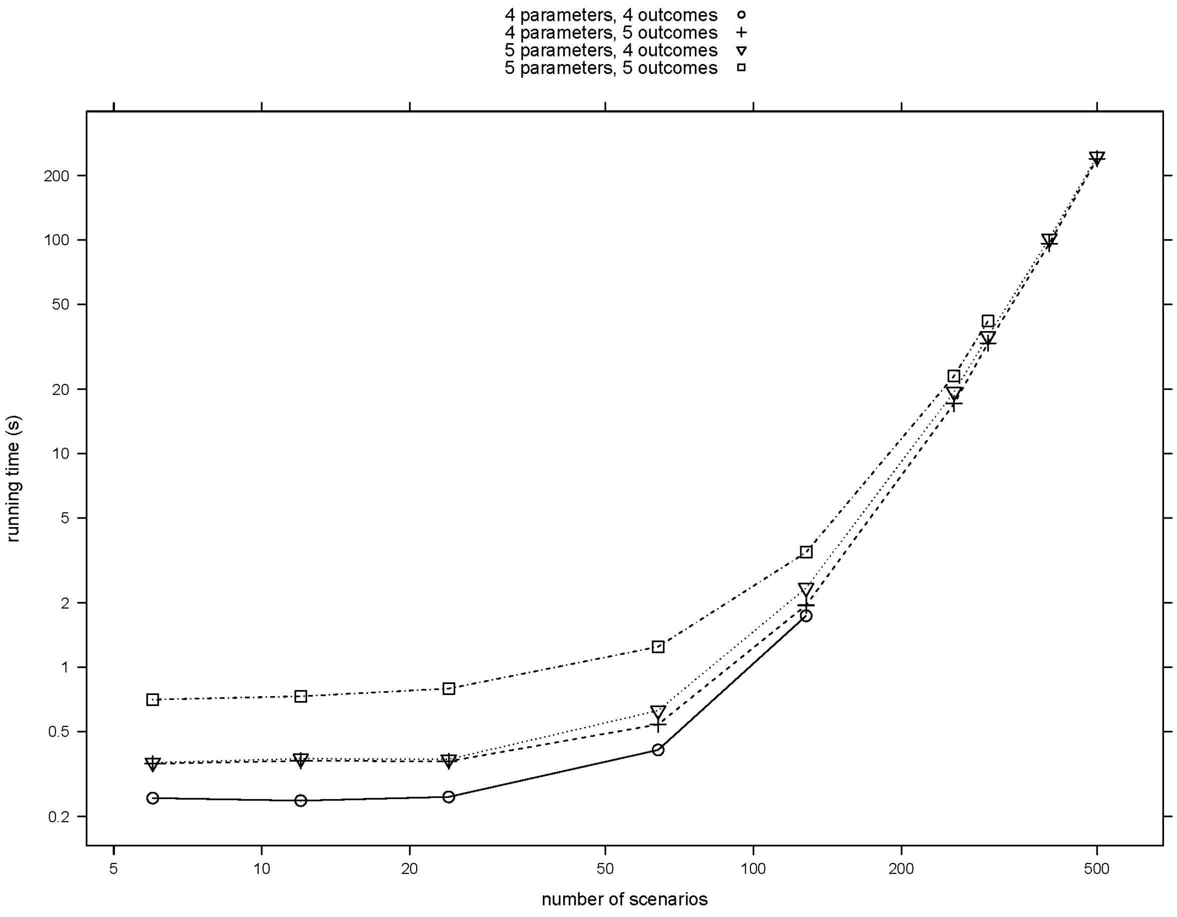


Figure 7.7 Average algorithm running times for the SNAC algorithm plotted against the number of scenarios.

Table 7.2 Summary of case studies used to test the SNAC algorithm

	Number of Uncertain Parameters								
	2		**3**		**4**			**5**	**6**
Possible Outcomes	4	10	4	5	3	4	5	4	4
Scenarios in Sample	12	24	6	24	12	128	24	64	512
Maximum Number of NACs	17	33	9	46	27	337	70	287	3338
Average Number of NACs	16.2	31.2	6.9	40.2	21.3	322.2	58.5	251.9	3119.7
NACs without Reductions	66	276	15	276	66	8,128	276	2,016	130,816

7.4 Conclusion

The premise of this paper was to develop an algorithm which would generate the minimum cardinality non-anticipativity constraint set for multistage stochastic programs where the uncertainty is endogenous and the realization of the endogenous parameter is gradual. The algorithm utilized knowledge of the subsets of scenarios formed by the realization of uncertainty in order to construct necessary and minimal sets of non-anticipativity constraints. An algorithm analysis showed that algorithm scaled in $O(S^3)$ as $S \rightarrow \infty$ as long as the number of uncertain parameters remain constant. Illustrative computational experiments confirm the scalability of the algorithm with respect to the number of scenarios and demonstrate significant reduction in the number of NACs generated.

CHAPTER 8
CONCLUSIONS AND FUTURE DIRECTIONS

8.1 Development of Heuristics for Planning and Scheduling

The development of heuristics for solving large-scale MSSPs has shown promise in providing tight feasible bounds. Both the KDA and MTSSP generate bounds for the clinical trial planning problem. Both heuristics provided feasible solutions for the problem within three percent of the optimal ENPV. The KDA was also extended to solve the NTIP problem. To do this several generalizations were made resulting in a new algorithm, EVDA, which includes a framework to solve any MSSP. The EVDA provided tight feasible solutions for the NTIP problem. Neither the EVDA nor the KDA excel at scheduling. Both represent greedy approaches when selecting variable values. Future work in this area should address the challenges of the scheduling under endogenous uncertainty. One potential approach would to be to consider an algorithm which combined both bottom-up and a top-down solution approaches. By weighing both the top-down solution and the bottom up solution, it may be possible that a hybrid solution may provide a solution with a better schedule.

8.2 Building Scalable Algorithms to Solve Multistage Stochastic Programs

The KDA algorithm explored in this dissertation is utilized as a primal bounding approach to solve the clinical trial planning problem. The development of the EVDA provides a framework for the generation of feasible bounds on MSSPs with non-trivial recourse action. The EVDA algorithm was capable of producing tight bounds on the NTIP problem. In future, the EVDA can be used as the primal bounding approach in the branch and bound algorithm. The branch and bound algorithm should then be applied to other MSSPs including the NTIP problem.

Branch and bound algorithms require both primal and dual bounding approaches. In the work presented in this dissertation, the dual bounding approaches considered were a modified progressive hedging approach and an optimal solution of each scenario (OSS) approach. The progressive hedging approach required the conversion of the MSSP with endogenous uncertainty (and exogenous realizations) to a MSSP with exogenous uncertainty. The resulting problem was separable but still required substantial computational resource to solve with the progressive hedging approach. In contrast, the OSS bounding approach was completely separable and resulted in a large number of very small problems. The downside to the OSS is that the dual bound showed very slow convergence. Many opportunities exist to explore approaches which are simple to solve similar to the OSS but provide tighter dual bounds. One such approach would be Lagrangean decomposition. The Lagrangean decomposition approach has been shown to be effective in solving MSSPs. (Gupta and Grossmann, 2011)

The branch and bound algorithm, particularly in its implementation on the clinical trial planning problem, spent many iterations evaluating solutions where the entire decision tree was delayed a single time period. In future, the development of a cutting approach which would remove non-optimal solutions may provide substantial increase in time required for the branch and bound algorithm to reach the optimal solution.

8.3 Structured Sampling for MSSP Approximation

The work in this dissertation developed a NAC generation algorithm for generating the minimum cardinality NAC set for MSSPs where the scenario set was a subset of the full set of scenarios. The use of the NAC generation algorithm is particularly useful when considering sample average approximation approaches for solving MSSPs. Sample average approximation uses a subset of scenarios randomly sampled from the full set to generate approximate solutions to the MSSP. In its limits, the sample average approximation converges to a dual bound on the MSSP. In future, an analysis of the impact of structured sampling approaches on the sample average approximation approach may be useful for the construction of tighter dual bounds as well as generating improved feasible solutions for the MSSP.

References

Apap, R., & Grossmann, I. E. (2015). Models and computational strategies for multistage stochastic programming under endogenous and exogenous uncertainties.

Apap, R. M., & Grossmann, I. E. (2016). Models and Computational Strategies for Multistage Stochastic Programming under Endogenous and Exogenous Uncertainties. *Computers & Chemical Engineering*. http://doi.org/http://dx.doi.org/10.1016/j.compchemeng.2016.11.011

Bayer. (2015). *Annual Report*.

Bellman, R. E. (1954). The Theory of Dynamic Programming. *Bulletin of the American Mathematical Society*. http://doi.org/10.1090/S0002-9904-1954-09848-8

Belotti, P., Lee, J., Liberti, L., Margot, F., & Wächter, A. (2009). Branching and bounds tighteningtechniques for non-convex MINLP. *Optimization Methods and Software*, *24*(4-5), 597–634. http://doi.org/10.1080/10556780903087124

Ben-Tal, A., El Ghaoui, L., & Nemirovski, A. (2009). Robust Optimization. *Princeton University Press*, *53*(3), 464–501. http://doi.org/10.1007/s10957-013-0421-6

Birge, J., & Louveaux, F. (2011). *Introduction to stochastic programming. Spriger, New York*. http://doi.org/10.1007/978-1-4614-0237-4

Blau, G. E., Pekny, J. F., Varma, V. A., & Bunch, P. R. (2004). Managing a Portfolio of Interdependent New Product Candidates in the Pharmaceutical Industry. *The Journal of Product Innovation Management*, *21*, 227–245.

Blau, G., Mehta, B., Bose, S., Pekny, J., Sinclair, G., Keunker, K., & Bunch, P. (2000). Risk management in the development of new products in highly regulated industries. *Computers & Chemical Engineering*, *24*, 659–664.

Boland, N., Dumitrescu, I., & Froyland, G. (2008). A Multistage Stochastic Programming Approach to Open Pit Mine Production Scheduling with Uncertain Geology. *Optimization Online*, 1–33.

Boland, N., Dumitrescu, I., Froyland, G., & Kalinowski, T. (2016). Minimum Cardinality Non-Anticipativity Constraint Sets for Multistage Stochastic Programming with Endogeneous Observation of Uncertainty. *Mathematical Programming: Series A and B*, *157*(1), 69–93.

Brown, D. B., & Caramanis, C. (2011). Robust Optimization. *SIAM Review*, *53*(3), 464–501.

Chen, Z. L., Li, S., & Tirupati, D. (2002). A scenario-based stochastic programming approach for technology and capacity planning. *Computers and Operations Research*, *29*(7), 781–806. http://doi.org/10.1016/S0305-0548(00)00076-9

Christian, B., & Cremaschi, S. (2015). Heuristic solution approaches to the pharmaceutical R&D pipeline management problem. *Computers and Chemical Engineering*, *74*, 34–47.

Christian, B., & Cremaschi, S. (2017). Variants to a knapsack decomposition heuristic for solving R&amp;D pipeline management problems. *Computers & Chemical Engineering*, *96*, 18–32. http://doi.org/10.1016/j.compchemeng.2016.10.011

Colvin, M., & Maravelias, C. T. (2008). A stochastic programming approach for clinical trial planning in new drug development. *Computers & Chemical Engineering*, *32*, 2626–2642. http://doi.org/10.1016/j.compchemeng.2007.11.010

Colvin, M., & Maravelias, C. T. (2009). Scheduling of testing tasks and resource planning in new product development using stochastic programming. *Computers & Chemical Engineering*, *33*, 964–976. http://doi.org/10.1016/j.compchemeng.2008.09.010

Colvin, M., & Maravelias, C. T. (2010). Modeling methods and a branch and cut algorithm for pharmaceutical clinical trial planning using stochastic programming. *European Journal of Operational Research*, *203*(1), 205–215. http://doi.org/10.1016/j.ejor.2009.07.022

Conforto, E. C., & Amaral, D. C. (2016). Agile project management and stage-gate model — A hybrid framework for technology-based companies. *Journal of Engineering and Technology Management*, *40*, 1–14. http://doi.org/10.1016/j.jengtecman.2016.02.003

Cooper, B. R. G. (2007). Managing Technology Development Projects. *IEEE Engineering Management Review*, *35*(1), 67–76.

Cooper, R. G. (1990). Stage-Gate Systems□: A New Tool for Managing New Products. *Business Horizons*, (33), 44–54.

Dal-mas, M., Giarola, S., Zamboni, A., & Bezzo, F. (2011). Strategic design and investment capacity planning of the ethanol supply chain under price uncertainty. *Biomass and Bioenergy*, *35*(5), 2059–2071. http://doi.org/10.1016/j.biombioe.2011.01.060

Dangl, T. (1999). Investment and capacity choice under uncertain demand. *European Journal of Operational Research*, *117*(3), 415–428.

Davis, G. A., & Owens, B. (2003). Optimizing the level of renewable electric R & D expenditures using real options analysis $. *Omega*, *31*, 1589–1608.

DuPont. (2015). *Annual Report*.

Eppen, G. D., Martin, R. K., & Schrage, L. (1989). OR Practice — A Scenario Approach to Capacity Planning. *Operations Research*, *37*(4), 517–527.

ExxonMobil. (2015). *Annual Report*.

Fahmi, I., & Cremaschi, S. (2012). Stage-Gate Representation of Feedstock Development for Chemical Process. In *Foundations of Computer-Aided Process Operations.*

Fahmi, I., & Cremaschi, S. (2013). A prototype simulation-based optimization approach to model feedstock development for chemical process industry. *Chemical Engineering Research and Design*, *91*(8), 1499–1507. http://doi.org/10.1016/j.cherd.2013.05.021

Fahmi, I., & Cremaschi, S. (2015). Global Solution Approaches for Biomass to Commodity Chemicals (BTCC) Investment Planning Problem. *CHEMICAL ENGINEERING TRANSACTIONS*, *43*, 1327–1332. http://doi.org/10.3303/CET1543222

Fahmi, I., Nuchitprasittichai, A., & Cremaschi, S. (2014). A new representation for modeling biomass to commodity chemicals development for chemical process industry. *Computers and Chemical Engineering*, *61*, 77–89. http://doi.org/10.1016/j.compchemeng.2013.10.012

Floudas, C. A., & Lin, X. (2005). Mixed Integer Linear Programming in Process Scheduling : Modeling , Algorithms , and Applications. *Annals of Operations Research*, *139*, 131–162.

Gardner, D. T., & Buzacott, J. A. (1999). Hedging against uncertainty in new technology development: The case of direct steelmaking. *IEEE Transactions on Engineering Management*, *46*(2), 177–189. http://doi.org/10.1109/17.759146

Goel, V., & Grossmann, I. E. (2004). A stochastic programming approach to planning of offshore gas field developments under uncertainty in reserves. *Computers and Chemical Engineering*, *28*(8), 1409–1429. http://doi.org/10.1016/j.compchemeng.2003.10.005

Goel, V., & Grossmann, I. E. (2006). A Class of Stochastic Programs with Decision Dependent Uncertainty. *Mathematical Programming*, *108*(2), 355–394.

Goel, V., Grossmann, I. E., El-bakry, A. S., & Mulkay, E. L. (2006). A novel branch and bound algorithm for optimal development of gas fields under uncertainty in reserves, *30*, 1076–1092. http://doi.org/10.1016/j.compchemeng.2006.02.006

Goyal, M., & Netessine, S. (2007). Strategic Technology Choice and Capacity Investment Under Demand Uncertainty. *Management Science*, *53*(2), 192–207. http://doi.org/10.1287/mnsc.1060.0611

Gupta, V., & Grossmann, I. E. (2011). Solution strategies for multistage stochastic programming with endogenous uncertainties. *Computers and Chemical Engineering*, *35*(11), 2235–2247. http://doi.org/10.1016/j.compchemeng.2010.11.013

Gupta, V., & Grossmann, I. E. (2014). Multistage stochastic programming approach for offshore oilfield infrastructure planning under production sharing agreements and endogenous uncertainties. *Journal of Petroleum Science and Engineering*, *124*, 180–197. http://doi.org/10.1016/j.petrol.2014.10.006

Hagspiel, V., Huisman, K. J. M., & Kort, P. M. (2016). Volume flexibility and capacity investment under demand uncertainty. *International Journal of Production Economics*, *178*, 95–108. http://doi.org/10.1016/j.ijpe.2016.05.007

Hart, W. E., Laird, C., Watson, J.-P., & Woodruff, D. L. (2012). *Pyomo–optimization modeling in python* (Vol. 67). Springer Science & Business Media.

Hooshmand, F., & Mirhassani, S. A. (2016). Efficient constraint reduction in multistage stochastic programming problems with endogenous uncertainty. *Optimization Methods and Software*, *31*(2), 359–376. http://doi.org/10.1080/10556788.2015.1088850

Kempf, K. G., Keskinocak, P., & Uzsoy, R. (2011). *Planning Production and Inventories in the Extended Enterprise* (Vol. 1). Springer.

Khaligh, F. H., & Mirhassani, S. A. (2015). A mathematical model for vehicle routing problem under endogenous uncertainty. *International Journal of Production Research*, *54*(2), 579–590. http://doi.org/10.1080/00207543.2015.1057625

Kostin, A. M., Guillén-gosálbez, G., Mele, F. D., Bagajewicz, M. J., & Jiménez, L. (2011). Chemical Engineering Research and Design Design and planning of infrastructures for bioethanol and sugar production under demand uncertainty. *Chemical Engineering Research and Design*, *90*(3), 359–376. http://doi.org/10.1016/j.cherd.2011.07.013

Kouvaritakis, N., Soria, A., & Isoard, S. (2004). Modelling energy technology dynamics: methodology for adaptive expectations models with learning by doing and learning by searching. *International Journal of Global Energy Issues (IJGEI)*, *14*(1-4), 104–115.

Krishnan, V., & Ulrich, K. T. (2001). Literature Product Development Decisions : A Review of the Literature. *Management Science*, *47*(1), 1–21.

Maravelias, C. T., & Grossmann, I. E. (2001). Simultaneous Planning for New Product Development and Batch Manufacturing Facilities. *Industrial and Engineering Chemistry Research*, *40*(26), 6147–6164.

Martínez-Costa, C., Mas-Machuca, M., Benedito, E., & Corominas, A. (2014). A review of mathematical programming models for strategic capacity planning in manufacturing. *International Journal of Production Economics*, *153*, 66–85. http://doi.org/10.1016/j.ijpe.2014.03.011

Medal, H. R., Pohl, E. A., Rossetti, M. D., Medal, H. R., Pohl, E. A., & Allocating, M. D. R. (2016). Allocating Protection Resources to Facilities When the Effect of Protection is Uncertain ABSTRACT. *IIE Transactions*, *48*(3), 220–234.

http://doi.org/10.1080/0740817X.2015.1078013

Mercier, L. (2009). *AMSAA: A Multistep Anticipatory Algorithm for Online Stochastic Combinatorial Optimization.*

Miremadi, M., Musso, C., & Oxgaard, J. (2013). McKinsey on Chemicals Chemical innovation : An investment for the ages. *McKinsey on Chemicals.*

Misener, R., Thompson, J. P., & Floudas, C. A. (2011). APOGEE: Global optimization of standard, generalized, and extended pooling problems via linear and logarithmic partitioning schemes. *Computers & Chemical Engineering*, *35*(5), 876–892. http://doi.org/10.1016/j.compchemeng.2011.01.026

Muhittin, O., & Ossama, K. (1992). A LINEARIZATION PROCEDURE FOR QUADRATIC AND CUBIC MIXED-INTEGER PROBLEMS. *Operations Research*, *40*(1), 109–117.

Mula, J., Poler, R., Garcı, J. P., & Lario, F. C. (2006). Models for production planning under uncertainty : A review $. *Internation Journal of Production Economics*, *103*(1), 271–285. http://doi.org/10.1016/j.ijpe.2005.09.001

Pekny, J. F. (2002). Algorithm architectures to support large-scale process systems engineering applications involving combinatorics , uncertainty , and risk management, *26*, 239–267.

Pietzsch, J. B., Shluzas, L. A., Paté-cornell, M. E., Yock, P. G., & Linehan, J. H. (2009). Stage-Gate Process for the Development of Medical Devices. *Transaction of the ASME*, *3*. http://doi.org/10.1115/1.3148836

PwC. (2012). From Vision to Decision - Pharma 2020. *Pwc Pharma 2020*, 56. Retrieved from http://www.pwc.com/gx/en/pharma-life-sciences/pharma2020/vision-to-decision.jhtml

Rockafellar, R. T., & Wets, R. J.-B. (1991). Scenarios and Policy Aggregation in Optimization Under Uncertainty. *Mathematics of Operations Research*, *16*(1), 119–147.

Rogers, M. J., Gupta, A., & Maranas, C. D. (2002). Real Options Based Analysis of Optimal Pharmaceutical Research and Development Portfolios. *Industrial and Engineering Chemistry Research*, *41*(25), 6607–6620.

Ross, S. M. (2006a). *Introduction to Probability Models*. Elsevier Science. Retrieved from https://books.google.com/books?id=0yDAZf1TfJEC

Ross, S. M. (2006b). *Introduction to Probability Models*. Elsevier Science.

Sahinidis, N. V. (2004). Optimization under uncertainty: State-of-the-art and opportunities. *Computers and Chemical Engineering*, *28*(6-7), 971–983. http://doi.org/10.1016/j.compchemeng.2003.09.017

Schmidt, C. W., & Grossmann, I. E. (1996). Optimization Models for the Scheduling of Testing Tasks in New Product Development. *Industrial and Engineering Chemistry Research*, *5885*(1995), 3498–3510.

Seider, W. D., Widagdo, S., Seader, J. D., & Lewin, D. R. (2009). Perspectives on chemical product and process design. *Computers and Chemical Engineering*, *33*(5), 930–935. http://doi.org/10.1016/j.compchemeng.2008.10.019

Solak, S., Clarke, J. P. B., Johnson, E. L., & Barnes, E. R. (2010). Optimization of R&D project portfolios under endogenous uncertainty. *European Journal of Operational Research*, *207*(1), 420–433. http://doi.org/10.1016/j.ejor.2010.04.032

Subramanian, D., Pekny, J. F., Reklaitis, G. V, & Blau, G. E. (2003). Simulation-Optimization Framework for Stochastic Optimization of R & D Pipeline Management. *AIChE Journal*, *49*(1), 96–112.

Subramanian, D., Pekny, J. F., & Reklaitis, G. V. (2001). A simulation-optimization framework for research and development pipeline management. *AIChE Journal*, *47*(10), 2226–2242. http://doi.org/10.1002/aic.690471010

Tarhan, B., & Grossmann, I. E. (2008). A multistage stochastic programming approach with strategies for uncertainty reduction in the synthesis of process networks with uncertain yields. *Computers & Chemical Engineering*, *32*(4-5), 766–788. http://doi.org/10.1016/j.compchemeng.2007.03.003

Tarhan, B., Grossmann, I. E., & Goel, V. (2008). A Multistage Stochastic Programming Approach for the Planning of Offshore Oil or Gas Field Infrastructure Under Decision Dependent Uncertainty, 1–61.

Tarhan, B., Grossmann, I. E., & Goel, V. (2009). Stochastic Programming Approach for the Planning of Offshore Oil or Gas Field Infrastructure under Decision-Dependent Uncertainty. *Industrial & Engineering Chemistry Research*, *48*(6), 3078–3097. http://doi.org/10.1021/ie8013549

Tarhan, B., Grossmann, I. E., & Goel, V. (2013). Computational strategies for non-convex multistage MINLP models with decision-dependent uncertainty and gradual uncertainty resolution. *Annals of Operations Research*, *203*(1), 141–166. http://doi.org/10.1007/s10479-011-0855-x

Tawarmalani, M., & Sahinidis, N. V. (2005). A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming*, *249*, 225–249.

Tingstro, J. (2006). Sustainability management in product development projects e the ABB experience. *Journal of Cleaner Production*, *14*, 1377–1385. http://doi.org/10.1016/j.jclepro.2005.11.027

Tsang, K. H., Samsatli, N. J., & Shah, N. (2007). Capacity Investment Planning for Multiple Vaccines Under Uncertainty: 1: Capacity Planning. *Food and Bioproducts Processing*, *85*(2), 120–128. http://doi.org/http://dx.doi.org/10.1205/fbp06001

Uytvanck, P. P. Van, Hallmark, B., Haire, G., Marshall, P. J., & Dennis, J. S. (2014). Impact of Biomass on Industry: Using Ethylene Derived from Bioethanol within the Polyester Value Chain.

Varma, V. A., Pekny, J. F., Blau, G. E., & Reklaitis, G. V. (2008). A framework for addressing stochastic and combinatorial aspects of scheduling and resource allocation in pharmaceutical R & D pipelines. *Computers and Chemical Engineering*, *32*, 1000–1015. http://doi.org/10.1016/j.compchemeng.2007.05.006

Varma, V. A., Uzsoy, R., & Pekny, J. (2007). Lagrangian heuristics for scheduling new product development projects in the pharmaceutical industry. *Journal of Heuristics*, *13*, 403–433. http://doi.org/10.1007/s10732-007-9016-4

Verderame, P. M., Elia, J. A., Li, J., & Floudas, C. A. (2010a). Planning and Scheduling under Uncertainty: A Review Across Multiple Sectors. *Industrial & Engineering Chemistry Research*, *49*(9), 3993–4017. http://doi.org/10.1021/ie902009k

Verderame, P. M., Elia, J. A., Li, J., & Floudas, C. A. (2010b). Planning and scheduling under uncertainty: A review across multiple sectors. *Industrial and Engineering Chemistry Research*, *49*(9), 3993–4017. http://doi.org/10.1021/ie902009k

Wang, J., & Hwang, W. (2007). A fuzzy set approach for R & D portfolio selection using a real options valuation model. *Omega*, *35*, 247–257. http://doi.org/10.1016/j.omega.2005.06.002

Watson, J. P., & Woodruff, D. L. (2011). Progressive hedging innovations for a class of stochastic mixed-integer resource allocation problems. *Computational Management Science*, *8*(4), 355–370. http://doi.org/10.1007/s10287-010-0125-4

Wright, T. P. (1936). Factors Affecting the Cost of Airplanes. *Journal of the Aeronautical Sciences*, *3*(4), 122–128. http://doi.org/10.2514/8.155

Yang, L., Wang, Y., Ma, J., To, C., & Cheng, T. C. E. (2014). Technology investment under flexible capacity strategy with demand uncertainty. *International Journal of Production Economics*, *154*, 190–197. http://doi.org/10.1016/j.ijpe.2014.04.008

Appendices

Appendix A Formulation of the Pharmaceutical R&D Pipeline Management Problem

The multistage stochastic programming formulation presented originally by Colvin and Maravelias (2008)

$$ENPV = \sum_{s} p_s(Rev_s + FRev_s - Cst_s) \tag{A.1}$$

$$Y_{d,j,t,s} = Y_{d,j,t-1,s} + X_{d,j,t-\tau_{i,j},s} \qquad \forall d,j,t,s \tag{A.2}$$

$$Z_{d,1,1,s} = 1 - X_{d,1,t,s} \qquad \forall d,s \tag{A.3}$$

$$Z_{d,1,t,s} = Z_{d,1,t-1,s} - X_{d,j,t,s} \qquad \forall d,t > 1,s \tag{A.4}$$

$$Z_{d,j,t,s} = Z_{d,j,t-1,s} + X_{d,j-1,t-\tau_{i,j-1},s} - X_{d,j,t,s} \qquad \forall d,\ j > 1,t,s \tag{A.5}$$

$$\sum_{t} X_{d,j,t,s} \leq 1 \qquad \forall d,j,s \tag{A.6}$$

$$\sum_{t\prime \leq t} X_{d,j,t',s} \leq Y_{d,j-1,t,s} \qquad \forall i,j > 1,t,s \tag{A.7}$$

$$\sum_{d} \sum_{j} \sum_{t'>t-\tau_{i,j}}^{t' \leq t} \rho_{d,j,r} X_{d,j,t',s} \leq \rho_r{}^{max}\ \forall r,t,s \tag{A.8}$$

$$X_{d,1,1,s} = X_{d,1,1,1} \qquad \forall i,s \tag{A.9}$$

$$-Y_{i^{s,s\prime},j^{s,s\prime},t,s} \leq X_{i,j,t,s} - X_{i,j,t,s'} \leq Y_{i^{s,s'},j^{s,s'},t,s} \qquad \forall i,j,\ (s,s') \in \Psi,\ t > 1 \tag{A.10}$$

$$Cst_s = \sum_{d,j,t} cd_t C_{d,j} X_{d,j,t,s} \quad \forall s \tag{A.11}$$

$$Rev_s = \sum_d \sum_t \Big\{ {rev_d}^{max} X_{d,PII,t,s} - {\gamma_d}^D \big(Z_{d,PII,t,s} + Z_{d,PIII,t,s} \big) - {\gamma_d}^L (t + \tau_{d,PIII}) X_{d,PIII,t,s} \Big\} \qquad \forall s \tag{A.12}$$

$$FRev_s = \sum_d \sum_j {rev_{d,j}}^{open} f_{d,j} Z_{d,j,|T|,s} + \sum_d \sum_{j\, \in \{PI,PII\}} \sum_{t > |T| - \tau_{d,j}} {rev_{d,j,t}}^{run} f_{d,j+1} X_{d,j,t,s} \qquad \forall s \tag{A.13}$$

$${rev_{d,j}}^{open} = {Rev_d}^{max} - {\gamma_d}^L \left(|T| + \sum_{j' \geq j} \tau_{d,j'} \right) \tag{A.14}$$

$${rev_{d,j}}^{run} = {Rev_d}^{max} - {\gamma_d}^L \left(t + \sum_{j' \geq j} \tau_{d,j'} \right) \tag{A.15}$$

$$f_{d,j} = 0.9 \left[\frac{{Rev_d}^{max} - {\gamma_d}^L |T| - \sum_{j' \geq j} C_{d,j'}}{{Rev_d}^{max} - {\gamma_d}^L |T|} \right] \tag{A.16}$$

Appendix B **The NTIP Problem**

Appendix B.1 **Linearization of $NF_{i,t,s}^{\alpha}$**

$$NF_{i,t,s}^{\alpha} = \left(1 - NN_{i,t,s}{}^{\alpha}\right)\left(\frac{RD_{i,t,s}}{RD_{i,0}}\right)^{\alpha_i{}^0} + NN_{i,t,s}{}^{\alpha}\left(\frac{RD_{i,t,s}}{RD_{i,0}}\right)^{\alpha_{i,s}{}^1} \tag{B.1}$$

$$NF_{i,t,s}^{\alpha} = \ 1 - NN_{i,t,s}{}^{\alpha} + NN_{i,t,s}{}^{\beta}\left(\frac{1}{RD_{i,0}}\right)^{\alpha_{i,s}^1}\left(RD_{i,t,s}\right)^{\alpha_{i,s}^1} \tag{B.2}$$

$$NF_{i,t,s}^{\alpha} = \ 1 - NN_{i,t,s}{}^{\alpha} + NN_{i,t,s}{}^{\beta}\left(\frac{1}{RD_{i,0}}\right)^{\alpha_{i,s}^1} RD\alpha1_{i,t,s} \tag{B.3}$$

$$a_i^{\alpha} = \frac{RD_i^{Max} - RD_i^{Min}}{3} \quad \forall i \tag{B.4}$$

$$\sum_{np\in\{1,2,3\}} b_{i,t,s,np}^{\alpha} = 1 \quad \forall i,t,s \tag{B.5}$$

$$\begin{aligned} RD_i^{Min} + \sum_{np\in\{1,2,3\}} a_i^{\alpha}\cdot(np-1)\cdot b_{i,t,s,np}^{\alpha} &\le RD_{i,t,s} \\ &\le RD_i^{Min} + \sum_{np\in\{1,2,3\}} a_i^{\alpha}\cdot np\cdot b_{i,t,s,np}^{\alpha} \quad \forall i,t,s \end{aligned} \tag{B.6}$$

$$\begin{aligned} RD\alpha1_{i,t,s} \ge \alpha_{i,s}^1\left(RD_{i,t,s}^{Min} + a_i^{\alpha}(np-1)\right)^{\alpha_{i,s}^1 - 1}\left(RD_{i,t,s} - \left(RD_{i,t,s}^{Min} + a_i^{\alpha}(np-1)\right)\right) \\ + \left(RD_{i,t,s}^{Min} + a_i^{\alpha}(np-1)\right)^{\alpha_{i,s}^1} \qquad \forall i,t,s,np\in\{1,2,3\} \end{aligned} \tag{B.7}$$

$$bRD_{i,t,s,np} \ \le b_{i,t,s,np} RD_{i,t,s}^{Max}\ \forall i,t,s \tag{B.8}$$

$$bRD_{i,t,s,np} \geq b_{i,t,s,np} RD_{i,t,s}^{Min} \quad \forall i,t,s \tag{B.9}$$

$$bRD_{i,t,s,np} \leq RD_{i,t,s} \quad \forall i,t,s \tag{B.10}$$

$$\begin{aligned} RD\alpha1_{i,t,s} \leq\ & bRD_{i,t,s,np} \frac{\left(RD_{i,t,s}^{Min} + a_i^{\alpha} \cdot np\right)^{\alpha_{i,s}^{1}} - \left(RD_{i,t,s}^{Min} + a_i^{\beta}(np-1)\right)^{\alpha_{i,s}^{1}}}{a_i^{\alpha}} \\ & - b_{i,t,s,np}^{\alpha}\left(RD_{i,t,s}^{Min} + a_i^{\alpha} \right. \\ & \left. \cdot np\right) \frac{\left(RD_{i,t,s}^{Min} + a_i^{\alpha} \cdot np\right)^{\alpha_{i,s}^{1}} - \left(RD_{i,t,s}^{Min} + a_i^{\beta}(np-1)\right)^{\alpha_{i,s}^{1}}}{a_i^{\alpha}} \\ & + \left(RD_{i,t,s}^{Min} + a_i^{\alpha} \cdot np\right)^{\alpha_{i,s}^{1}} \forall i,t,s \end{aligned} \tag{B.11}$$

$$NF_{i,t,s}^{\alpha} = \ 1 - NN_{i,t,s}^{\alpha} + \left(\frac{1}{RD_{i,0}}\right)^{\alpha_{i,s}^{1}} NRD\alpha1_{i,t,s} \quad \forall i,t,s \tag{B.12}$$

$$NRD\alpha1_{i,t,s} \leq NN_{i,t,s}{}^{\alpha} \cdot RD\alpha1_{i,t,s}^{Max} \quad \forall i,t,s \tag{B.13}$$

$$NRD\alpha1_{i,t,s} \geq NN_{i,t,s}{}^{\alpha} \cdot RD\alpha1_{i,t,s}^{Min} \quad \forall i,t,s \tag{B.14}$$

$$NRD\alpha1_{i,t,s} \leq RD\alpha1_{i,t,s} \quad \forall i,t,s \tag{B.15}$$

Appendix B.2 Linearization of $\boldsymbol{CC_{i,0}NF^{\beta}_{i,t,s}NF^{\alpha}_{i,t,s}}$

$$CC_{i,t,s} = C_{i,0}NF\beta\alpha_{i,t,s} \quad \forall i,t,s \tag{B.16}$$

$$a_{NF^{\beta}} = \frac{NF^{\beta,Max}_{i,t,s} - NF^{\beta,Min}_{i,t,s}}{3} \tag{B.17}$$

$$\sum_{np\in\{1,2,3\}} b^{NF\beta}_{i,t,s,np} = 1 \quad \forall i,t,s \tag{B.18}$$

$$\begin{aligned} NF^{\beta,Min}_{i,t,s} + \sum_{np\in\{1,2,3\}} a_{NF^{\beta}} \cdot (np-1) \cdot b^{NF\beta}_{i,t,s,np} &\le NF^{\beta}_{i,t,s} \\ &\le NF^{\beta,Min}_{i,t,s} + \sum_{np\in\{1,2,3\}} a_{NF^{\beta}} \cdot np \cdot b^{NF\beta}_{i,t,s,np} \quad \forall i,t,s \end{aligned} \tag{B.19}$$

$$NF^{\alpha}_{i,t,s} = NF^{\alpha,Min}_{i,t,s} + \sum_{np\in\{1,2,3\}} \Delta NF^{\alpha}_{i,t,s,np} \quad \forall i,t,s \tag{B.20}$$

$$0 \le \Delta NF^{\alpha}_{i,t,s,np} \le \left(NF^{\alpha,Max}_{i,t,s} - NF^{\alpha,Min}_{i,t,s}\right) \cdot b^{NF\beta}_{i,t,s,np} \quad \forall np \in \{1,2,3\} \tag{B.21}$$

$$\begin{aligned} NF\beta\alpha_{i,t,s} \ge\ & NF^{\beta}_{i,t,s} \cdot NF^{\alpha,Min}_{i,t,s} \\ & + \sum_{np\in\{1,2,3\}} \left(NF^{\beta,Min}_{i,t,s} + a_{NF^{\beta}} \cdot (np-1)\right) \cdot \Delta NF^{\alpha}_{i,t,s,np} \quad \forall i,t,s \end{aligned} \tag{B.22}$$

$$\begin{aligned} NF\beta\alpha_{i,t,s} \ge\ & NF^{\beta}_{i,t,s} \cdot NF^{\alpha,Max}_{i,t,s} \\ & + \sum_{np\in\{1,2,3\}} \left(NF^{\beta,Min}_{i,t,s} + a_{NF^{\beta}} \cdot np\right) \cdot \Big(\Delta NF^{\alpha}_{i,t,s,np} \\ & - \left(NF^{\alpha,Max}_{i,t,s} - NF^{\alpha,Min}_{i,t,s}\right) \cdot b^{NF\beta}_{i,t,s,np}\Big) \quad \forall i,t,s \end{aligned} \tag{B.23}$$

$$NF\beta\alpha_{i,t,s} \leq NF_{i,t,s}^{\beta} \cdot NF_{i,t,s}^{\alpha,Min} + \sum_{np\in\{1,2,3\}} \left(NF_{i,t,s}^{\beta,Min} + a_{NF^{\beta}} \cdot np\right) \cdot \Delta NF_{i,t,s,np}^{\alpha} \quad \forall i,t,s \tag{B.24}$$

$$NF\beta\alpha_{i,t,s} \leq NF_{i,t,s}^{\beta} \cdot NF_{i,t,s}^{\alpha,Max} + \sum_{np} \left(NF_{i,t,s}^{\beta,Min} + a_{NF^{\beta}} \cdot (np-1)\right) \cdot \left(\Delta NF_{i,t,s,np}^{\alpha} - \left(NF_{i,t,s}^{\alpha,Max} - NF_{i,t,s}^{\alpha,Min}\right) \cdot b_{i,t,s,np}^{NF\beta}\right) \quad \forall i,t,s \tag{B.25}$$

Appendix B.3 **Case Study 2**

Table B.1 Uncertain parameter realizations for Case Study 2

Technology	α Values		β Values		χ Values	
	High	Low	High	Low	High	Low
TECH1	-0.19	-0.14	-0.09	-0.08	0.96	0.93
TECH2	-0.18	-0.20	-0.08	-0.06	0.98	0.95

Table B.2 Probability of a technology completing each stage

Technology	Probability of Project Success	
	Laboratory	Pilot Plant
TECH1	98%	99%
TECH2	95%	99%

Table B.3 Demand Realizations for Case Study 2

Chemcial	Demand (Mtonnes)			
	2		3	
	High	Low	High	Low
CHEM1	0		0	
CHEM2	0		0	
CHEM5	23.5	28.8	22.1	26

Table B.4 Case Study 2 Technology specific fixed parameters

	Technology	
	TECH1	TECH2
MaximumCapacity Expansion (Mtonnes)	6	6
Initial R&D Investment (Trillion Dollars)	1	5
Initial Installed Capacity (Mtonnes)	1.0	2.5
Initial Capacity Expansion Cost ($/kg)	1.0	1.4

Table B.5 Case Study 2 Chemical specific fixed Parameters

Chemcial	Initial Cost ($/tonne)	Molecular Weight (kg/kmol)
CHEM1	742	50
CHEM2	845.00	62
CHEM3	1200	72

Appendix B.4 **Case Study 3**

Table B.6 Uncertain parameter realizations for Case Study 3

Technology	α Values		β Values		χ Values	
	High	Low	High	Low	High	Low
TECH1	0		0		0.60	
TECH2	-0.17	-0.15	-0.10	-0.09	0.96	0.91
TECH3	0		0		0	
TECH4	-0.21	-0.13	-0.09	-0.07	0.98	0.94

Table B.7 Demand Realizations for Case Study 3

Chemcial	Demand (Mtonnes)					
	2		3		4	
	High	Low	High	Low	High	Low
CHEM1	0		0		0	
CHEM2	0		0		0	
CHEM3	0		0		0	
CHEM4	0		0		0	
CHEM5	28	23.5	26	22.1	25	23.6

Table B.8 Probability of a technology completing each stage

Technology	Probability of Project Success	
	Laboratory	Pilot Plant
TECH1	N/A	
TECH2	58%	96%
TECH3	N/A	
TECH4	63%	91%

Table B.9 Chemical specific fixed parameters for Case Study 3

Chemcial	Initial Cost ($/tonne)	Molecular Weight (kg/kmol)
CHEM1	500	60
CHEM2	800	100
CHEM3	900	51
CHEM4	1100	62
CHEM5	1200	72

Table B.10 Technology specific fixed parameters for Case Study 3

	Technology			
	TECH1	TECH2	TECH3	TECH4
Maximum Capacity Expansion (Mtonnes)	6	6	6	6
Initial R&D Investment (Trillion Dollars)	1	1	1	1
Initial Installed Capacity (Mtonnes)	22	2.5 e-7	34	1e-7
Initial Capacity Expansion Cost ($/kg)	1.0	6	0.4	10

Appendix B.5 **Biomass To Commodity Chemical**

Table B.11 Values for the different realizations of α,β, and χ, and Probabilities of Success

Technology	α Values		β Values		χ Values	
	High	Low	High	Low	High	Low
Gasification	-.021	-0.19	-0.05	-0.07	0.95	0.92
Catalytic Conversion	-0.19	-0.21	-0.08	-0.06	0.35	0.46
Fermentation	0		0			
Catalytic Dehydration	0		0		0.55	
Cracking	0		0		0.45	

Table B.12 Probability of a technology completing each stage

Technology	Probability of Project Success	
	Laboratory	Pilot Plant
Gasification	75%	88%
Catalytic Conversion	89%	90%
Fermentation	N/A	
Catalytic Dehydration	N/A	
Cracking	N/A	

Table B.13 Demand Realizations for Case Study 3

Chemcial	Demand (Mtonnes)			
	2		3	
	High	Low	High	Low
Biomass	0		0	
Syngas	0		0	
Naphtha	0		0	
Ethanol	0		0	
Ethylene	28	23.5	26	22.1

Table B.14 Chemical specific fixed parameters for Case Study 3

Chemcial	Initial Cost ($/tonne)	Molecular Weight (kg/kmol)
Biomass	150	342.3
Syngas	250	30.28
Naphtha	350	99
Ethanol	837	46.07
Ethylene	1500	28.05

Table B.15 Technology specific fixed parameters for Case Study 3

	Technology				
	Gasification	Cracking	Fermentation	Catalytic Conversion	Catalytic Dehydration
Maximum Capacity Expansion (Mtonnes)	6	6	6	6	6
Initial R&D Investment (Trillion Dollars)	1	1	1	1	1
Initial Installed Capacity (Mtonnes)	1e-7	18	1e-4	1e-7	1e-4
Initial Capacity Expansion Cost ($/kg)	10	1.2	1.4	1	10

Appendix C **Parameters for Base Case Problems**

Table C.1 Parameters for the Two-Product Base Case (2_2_5_2)

	Duration		**Probability of Success**		**Cost($1M)**		**Resource 1 (max =2)**		**Resource 2 (max=3)**		rev^{max}	Y^L	Y^D
Product	PI	PII	PI	PII	PI	PII	PI	PII	PI	PII			
D1	2	4	0.3	0.5	10	90	1	1	1	2	3100	19.2	44
D2	2	3	0.4	0.6	10	80	1	2	1	1	3250	19.6	56

Table C.2 Parameters for the Three-Product Base Case (3_3_12_2)

	Duration			**Probability of Success**			**Trial Cost ($M)**			**Resource 1 (max =2)**			**Resource 1 (max =3)**			rev^{max}	Y^L	Y^D
Product	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII			
D1	2	4	4	0.3	0.5	0.8	10	90	220	1	1	2	1	2	3	3100	19.2	22
D2	2	3	5	0.4	0.6	0.8	10	80	200	1	2	2	1	1	3	3250	19.6	28
D3	2	3	4	0.3	0.6	0.9	10	90	180	1	1	2	1	1	3	3300	20	26

Table C.3 Parameters for the Four-Product Base Case (4_3_6_2)

	Duration			**Probability of Success**			**Trial Cost ($M)**			**Resource 1 (max =4)**			**Resource 2 (max =3)**			rev^{max}	Y^L	Y^D
Product	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII			
D1	1	1	3	0.3	0.5	0.8	10	90	220	1	1	2	1	2	3	3100	19.2	22
D2	1	2	2	0.4	0.6	0.8	10	80	200	1	2	2	1	1	3	3250	19.6	28
D3	1	1	3	0.3	0.6	0.9	10	90	180	1	1	2	1	1	3	3300	20	26
D4	1	2	2	0.4	0.6	0.8	10	100	170	1	1	2	1	2	3	3000	19.4	24

Table C.4 Parameters for the Five-Product Base Case (4_3_6_2)

	Duration			Probability of Success			Trial Cost ($M)			Resource 1 (max =4)			Resource 2 (max =3)			rev^{max}	Y^L	Y^D
Product	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII			
D1	1	1	3	0.3	0.5	0.8	10	90	220	1	1	2	1	2	3	3100	19.2	22
D2	1	2	2	0.4	0.6	0.8	10	80	200	1	2	2	1	1	3	3250	19.6	28
D3	1	1	3	0.3	0.6	0.9	10	90	180	1	1	2	1	1	3	3300	20	26
D4	1	2	2	0.4	0.6	0.8	10	100	170	1	1	2	1	2	3	3000	19.4	24
D5	1	2	3	0.35	0.5	0.9	10	70	210	1	1	2	1	1	3	3150	19.6	24

Table C.5 Parameters for the Six-Product Base Case (6_3_6_2)

	Duration			Probability of Success			Trial Cost ($M)			Resource 1 (max =4)			Resource 2 (max =3)			rev^{max}	Y^L	Y^D
Product	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII	PI	PII	PIII			
D1	1	1	3	0.3	0.5	0.8	10	90	220	1	1	2	1	2	3	3100	19.2	22
D2	1	2	2	0.4	0.6	0.8	10	80	200	1	2	2	1	1	3	3250	19.6	28
D3	1	1	3	0.3	0.6	0.9	10	90	180	1	1	2	1	1	3	3300	20	26
D4	1	2	2	0.4	0.6	0.8	10	100	170	1	1	2	1	2	3	3000	19.4	24
D5	1	2	3	0.35	0.5	0.9	10	70	210	1	1	2	1	1	3	3150	19.6	24
D6	1	2	3	0.45	0.45	0.8	10	85	195	1	2	2	2	1	3	3050	19	25

Appendix D Computational Results for Parameter and Size Variations

Table D.1 Parameter and Size Perturbation Results for the Two-Product Base Case (OP)

File Name	ENPV	Percent Difference (MSSP)	Percent Difference (OP)	Same as Base KDA solution?	Knapsack Problem Count	KDA Solve Time	MSSP ENPV	MSSP Total Time
2_2_5_3	1097	-0.63	0.00	TRUE	4	0.07	1104	0.15
2_2_5_4	1097	-0.63	0.00	TRUE	4	0.07	1104	0.16
2_2_5_5	1097	-0.63	0.00	TRUE	4	0.07	1104	0.19
2_3_5_2	709	-3.26	-35.36	FALSE	4	0.07	733	0.33
2_4_5_2	523	-5.72	-52.30	FALSE	4	0.07	555	0.58
2_5_5_2	422	-5.03	-61.56	FALSE	5	0.09	444	1.15
2_2_4_2	1101	0.19	0.35	FALSE	4	0.07	1099	0.11
2_2_6_2	1110	0.00	1.17	TRUE	6	0.09	1110	0.15
2_2_7_2	1110	-0.10	1.17	TRUE	6	0.11	1111	0.19
2_2_8_2	1110	-0.39	1.17	TRUE	6	0.11	1114	0.21
2_2_5_2_C.75	1115	-0.52	1.62	TRUE	4	0.08	1121	0.24
2_2_5_2_C.9	1104	-0.59	0.65	TRUE	4	0.07	1111	0.14
2_2_5_2_C1.1	1090	-0.68	-0.65	TRUE	4	0.08	1098	0.13
2_2_5_2_C1.25	1079	-0.75	-1.62	TRUE	4	0.08	1088	0.14
2_2_5_2_R.75	791	-0.88	-27.94	TRUE	4	0.06	798	0.13
2_2_5_2_R.9	975	-0.71	-11.18	TRUE	4	0.08	982	0.16
2_2_5_2_R1.1	1220	-0.57	11.18	TRUE	4	0.07	1227	0.16
2_2_5_2_R1.25	1404	-0.49	27.94	TRUE	4	0.08	1411	0.15
2_2_5_2_GL.75	1110	-0.75	1.14	TRUE	4	0.08	1118	0.15
2_2_5_2_GL.9	1102	-0.68	0.46	TRUE	4	0.08	1110	0.15
2_2_5_2_GL1.1	1092	-0.59	-0.46	TRUE	4	0.07	1099	0.15
2_2_5_2_GL1.25	1085	-0.52	-1.14	TRUE	4	0.08	1090	0.14
2_2_5_2_GD.75	1099	-0.45	0.18	TRUE	4	0.16	1104	0.15
2_2_5_2_GD.9	1098	-0.56	0.07	TRUE	4	0.08	1104	0.15
2_2_5_2_GD1.1	1096	-0.71	-0.07	TRUE	4	0.08	1104	0.14
2_2_5_2_GD1.25	1095	-0.81	-0.18	TRUE	4	0.07	1104	0.15
2_2_5_2_TD1	1087	0.87	-0.89	FALSE	4	0.03	1078	0.14
2_2_5_2_TD2	1077	0.72	-1.81	FALSE	4	0.03	1070	0.14
2_2_5_2_PC60	1097	-0.63	0.00	TRUE	4	0.08	1104	0.15
2_2_5_2_PC30	1097	-0.63	0.00	TRUE	4	0.08	1104	0.15
2_2_5_2_PC0	1121	0.00	2.13	FALSE	4	0.08	1121	0.15
Average		***-0.84***						

Table D.2 Parameter and Size Perturbation Results for the Three-Product Base Case (OP)

File Name	ENPV	Percent Difference (MSSP)	Percent Difference (OP)	Same as Base KDA solution?	Knapsack Problem Count	KDA Solve Time	MSSP ENPV	MSSP Total Time
3_3_12_3	1180	-0.67	0.17	FALSE	21	0.46	1188	5.44
3_3_12_4	1180	-0.67	0.17	FALSE	21	0.43	1188	5.82
3_3_12_5	1180	-0.79	0.17	FALSE	21	0.48	1189	5.88
3_4_12_2	761	-1.60	-35.38	FALSE	21	0.75	773	16.44
3_5_12_2	417	-15.29	-64.60	FALSE	21	0.51	492	44.90
3_6_12_2	240	-22.10	-79.64	FALSE	31	0.94	308	109.61
3_3_11_2	1177	-0.92	-0.04	FALSE	20	0.78	1188	4.68
3_3_13_2	1174	-1.24	-0.31	FALSE	27	1.07	1189	5.68
3_3_14_2	1178	-1.25	0.02	FALSE	28	1.10	1193	10.16
3_3_15_2	1177	-1.68	-0.10	FALSE	33	1.41	1197	6.46
3_3_12_2_C.75	1225	-0.92	4.03	FALSE	23	0.61	1237	5.39
3_3_12_2_C.9	1197	-0.96	1.61	FALSE	23	0.58	1208	5.46
3_3_12_2_C1.1	1159	-1.01	-1.61	FALSE	23	0.58	1171	5.32
3_3_12_2_C1.25	1130	-1.00	-4.03	FALSE	23	0.58	1142	5.35
3_3_12_2_R.75	804	-1.11	-31.76	FALSE	23	0.61	813	5.34
3_3_12_2_R.9	1028	-1.03	-12.71	FALSE	23	0.55	1039	5.25
3_3_12_2_R1.1	1327	-1.00	12.71	FALSE	23	0.57	1341	5.40
3_3_12_2_R1.25	1552	-0.92	31.79	FALSE	23	0.56	1567	5.20
3_3_12_2_GL.75	1209	-0.96	2.67	FALSE	23	0.59	1221	5.28
3_3_12_2_GL.9	1190	-0.99	1.07	FALSE	23	0.60	1202	5.36
3_3_12_2_GL1.1	1165	-1.04	-1.07	FALSE	23	0.60	1177	5.24
3_3_12_2_GL1.25	1146	-1.11	-2.67	FALSE	23	0.55	1159	5.42
3_3_12_2_GD.75	1179	-1.02	0.08	FALSE	23	0.57	1191	5.28
3_3_12_2_GD.9	1178	-1.02	0.03	FALSE	23	0.57	1190	5.28
3_3_12_2_GD1.1	1177	-1.02	-0.03	FALSE	23	0.58	1189	5.40
3_3_12_2_GD1.25	1177	-1.02	-0.08	FALSE	23	0.58	1189	5.28
3_3_12_2_TD1	1132	-1.85	-3.95	FALSE	13	0.31	1153	5.60
3_3_12_2_TD2	1083	-2.39	-8.09	FALSE	7	0.28	1109	5.66
3_3_12_2_PC60	1183	-1.61	0.48	FALSE	33	0.82	1203	5.28
3_3_12_2_PC30	1213	-0.58	2.96	FALSE	31	0.77	1220	5.37
3_3_12_2_PC0	1214	-0.59	3.09	FALSE	27	0.67	1221	5.27
Average		***-2.24***						

Table D.3 Parameter and Size Perturbation Results for the Four-Product Base Case (OP)

File Name	ENPV	Percent Difference (MSSP)	Percent Difference (OP)	Same as Base KDA solution?	Knapsack Problem Count	KDA Solve Time	MSSP ENPV	MSSP Total Time
4_3_6_3	1619	-2.63	-3.39	FALSE	63	1.65	1663	22.76
4_3_6_4	1619	-2.25	-3.39	FALSE	63	1.90	1656	23.58
4_3_6_5	1624	-1.67	-3.09	FALSE	49	1.43	1651	24.40
4_4_6_2	1095	-3.64	-34.65	FALSE	50	1.42	1136	126.59
4_5_6_2	687	-13.17	-58.99	FALSE	24	0.93	791	554.35
4_6_6_2	405	-22.59	-75.80	FALSE	24	1.35	524	2067.58
4_3_5_2	1664	-0.73	-0.67	FALSE	32	2.30	1677	17.83
4_3_7_2	1687	-0.01	0.67	FALSE	80	5.46	1687	25.58
4_3_8_2	1695	0.10	1.15	FALSE	111	7.74	1693	29.49
4_3_9_2	1699	-0.01	1.39	FALSE	139	9.73	1699	59.11
4_3_6_2_C.75	1747	-0.66	4.24	FALSE	54	3.20	1758	24.45
4_3_6_2_C.9	1704	-0.67	1.70	FALSE	54	3.13	1716	23.45
4_3_6_2_C1.1	1647	-0.37	-1.70	FALSE	54	2.84	1653	24.05
4_3_6_2_C1.25	1605	-0.54	-4.24	FALSE	54	2.18	1613	23.53
4_3_6_2_R.75	1160	-0.86	-30.75	FALSE	54	2.14	1170	23.48
4_3_6_2_R.9	1469	-1.07	-12.30	FALSE	54	2.06	1485	23.48
4_3_6_2_R1.1	1882	-0.59	12.30	FALSE	54	2.06	1893	23.55
4_3_6_2_R1.25	2191	-0.70	30.76	FALSE	54	2.05	2206	23.29
4_3_6_2_GL.75	1699	-0.84	1.42	FALSE	54	2.07	1714	23.46
4_3_6_2_GL.9	1685	-0.54	0.57	FALSE	54	2.02	1694	23.70
4_3_6_2_GL1.1	1666	-0.77	-0.57	FALSE	54	2.04	1679	23.62
4_3_6_2_GL1.25	1652	-0.79	-1.42	FALSE	54	2.04	1665	23.46
4_3_6_2_GD.75	1677	-0.68	0.10	FALSE	54	1.98	1689	23.55
4_3_6_2_GD.9	1676	-0.78	0.04	FALSE	54	1.99	1689	23.38
4_3_6_2_GD1.1	1675	-0.51	-0.04	FALSE	54	1.91	1684	23.63
4_3_6_2_GD1.25	1674	-0.56	-0.10	FALSE	54	1.88	1683	23.48
4_3_6_2_TD1	1602	-2.04	-4.40	FALSE	19	0.21	1636	24.09
4_3_6_2_TD2	1486	-1.55	-11.32	FALSE	18	0.21	1510	23.86
4_3_6_2_PC60	1676	-1.57	0.02	FALSE	85	3.10	1703	23.66
4_3_6_2_PC30	1706	-0.74	1.79	FALSE	81	2.97	1718	23.23
4_3_6_2_PC0	1714	-0.40	2.31	FALSE	81	3.02	1721	23.46
Average		***-2.06***						

Table D.3 Parameter and Size Perturbation Results for the Five-Product Base Case (OP)

File Name	ENPV	Percent Difference (MSSP)	Percent Difference (OP)	Same as Base KDA solution?	Knapsack Problem Count	KDA Solve Time	MSSP ENPV	MSSP Total Time
5_3_6_3	1982	-2.82	-2.95	FALSE	90	1.96	2040	286.94
5_3_6_4	1982	-2.62	-2.95	FALSE	90	3.81	2036	294.54
5_3_6_5	1994	-1.90	-2.35	FALSE	87	4.11	2033	296.37
5_4_6_2	1323	-3.77	-35.24	FALSE	68	3.35	1375	2963.13
5_5_6_2	860	-10.83	-57.91	FALSE	26	2.28	964	21376.09
5_6_6_2	509	-18.86	-75.08	FALSE	26	0.61	627	112356.45
5_3_5_2	2038	0.11	-0.25	FALSE	51	1.10	2035	227.08
5_3_7_2	2059	-0.10	0.79	FALSE	147	3.10	2061	321.21
5_3_8_2	2064	0.03	1.05	FALSE	194	4.15	2063	366.56
5_3_9_2	2084	0.44	2.01	FALSE	293	6.32	2075	414.11
5_3_6_2_C.75	2128	-0.28	4.18	FALSE	75	1.59	2134	278.36
5_3_6_2_C.9	2077	-0.25	1.67	FALSE	75	1.60	2082	278.04
5_3_6_2_C1.1	2008	-0.50	-1.67	FALSE	75	1.60	2019	851.48
5_3_6_2_C1.25	1957	-0.54	-4.18	FALSE	75	1.55	1968	313.37
5_3_6_2_R.75	1413	-0.30	-30.80	FALSE	75	1.61	1418	288.00
5_3_6_2_R.9	1791	-0.41	-12.32	FALSE	75	1.67	1798	280.77
5_3_6_2_R1.1	2294	-0.15	12.32	FALSE	75	1.62	2298	280.53
5_3_6_2_R1.25	2672	-0.66	30.82	FALSE	75	1.62	2690	279.43
5_3_6_2_GL.75	2074	-0.47	1.53	FALSE	75	1.61	2084	279.67
5_3_6_2_GL.9	2055	-0.31	0.61	FALSE	75	1.62	2061	284.72
5_3_6_2_GL1.1	2030	-0.77	-0.61	FALSE	75	1.63	2046	282.60
5_3_6_2_GL1.25	2011	-0.65	-1.53	FALSE	75	1.65	2025	283.77
5_3_6_2_GD.75	2045	-1.20	0.11	FALSE	75	1.58	2069	280.57
5_3_6_2_GD.9	2043	-0.84	0.04	FALSE	75	1.56	2061	287.98
5_3_6_2_GD1.1	2042	-0.19	-0.04	FALSE	75	1.57	2046	280.94
5_3_6_2_GD1.25	2040	-0.37	-0.11	FALSE	75	1.65	2048	285.80
5_3_6_2_TD1	1957	-2.32	-4.20	FALSE	21	0.12	2004	279.89
5_3_6_2_TD2	1840	-1.22	-9.96	FALSE	5	0.25	1862	286.54
5_3_6_2_PC60	2097	-1.10	2.68	FALSE	215	9.72	2121	282.39
5_3_6_2_PC30	2118	-0.45	3.68	FALSE	243	11.21	2127	280.09
5_3_6_2_PC0	2120	-0.39	3.78	FALSE	243	11.28	2128	282.89
Average		***-1.73***						

Appendix E Knapsack Variation Case Study Information

Table E.8.1 Parameter variation values from base case values

Parameter Varied	**Variation**			
Trial Cost	-25%	-10%	+10%	+25%
Revenue	-25%	-10%	+10%	+25%
Gamma-L	-25%	-10%	+10%	+25%
Gamma-D	-25%	-10%	+10%	+25%
Percent Constrained	30%	60%	100%	
Planning Horizon Length	-1	+1	+2	+3
Number of Trials	+1	+2	+3	
Number of Resources	+1	+2	+3	

Table E.8.2 Number of resources required for each trial when +1,+2 and +3 resources are added for each

	PI			**PII**			**PIII**		
Base Case	**+1**	**+2**	**+3**	**+1**	**+2**	**+3**	**+1**	**+2**	**+3**
Two-Product									
Product1	2	0	0	1	1	2	N/A	N/A	N/A
Product2	1	0	2	2	1	2	N/A	N/A	N/A
Three-Product									
Product 1	2	0	0	1	1	2	1	4	3
Product 2	1	0	2	3	0	3	1	3	3
Product 3	0	0	1	2	1	0	3	3	2
Four-Product									
Product 1	2	0	0	1	1	2	1	4	3
Product 2	1	0	2	3	0	3	1	3	3
Product 3	0	0	1	2	1	0	3	3	2
Product 4	1	2	1	0	0	3	3	4	2
Five Product									
Product 1	2	0	0	1	1	2	1	4	3
Product 2	1	0	2	3	0	3	1	3	3
Product 3	0	0	1	2	1	0	3	3	2
Product 4	1	2	1	0	0	3	3	4	2
Product 5	1	2	2	0	0	0	3	2	1

Appendix F SNAC Algorithm Case Studies

Number of Uncertain Parameters	Number of Realizations for Each Uncertain Parameter	Number of Scenarios Sampled	30 Iteration SNAC Time (CPU Seconds)	Average Sample SNAC Time (CPU Seconds)	NAC Count (Average)	NAC Count (Maximum)
2	4	6	4.73	0.16	5.57	7
2	4	12	4.96	0.17	16.20	17
2	10	6	18.60	0.62	5.10	6
2	10	12	18.74	0.62	12.30	15
2	10	24	19.34	0.64	31.17	33
2	10	64	25.78	0.86	108.17	109
3	5	6	11.76	0.39	7.27	11
3	5	12	12.05	0.40	17.43	20
3	5	24	12.16	0.41	40.23	46
3	5	64	19.14	0.64	128.03	133
4	3	6	7.43	0.25	7.67	12
4	3	12	7.65	0.25	21.33	27
4	3	24	8.01	0.27	49.90	61
4	3	64	15.22	0.51	152.20	157
4	4	6	12.79	0.43	7.93	10
4	4	12	12.96	0.43	22.70	31
4	4	24	12.82	0.43	53.60	66
4	4	64	20.81	0.69	159.37	178
4	4	128	74.44	2.48	322.20	337
4	4	256	706.71	23.56	768.00	768
4	5	6	21.17	0.71	7.90	11
4	5	12	20.59	0.69	23.13	29
4	5	24	21.26	0.71	58.53	70
4	5	64	31.82	1.06	179.27	201
4	5	128	91.12	3.04	352.10	385
4	5	256	700.00	23.33	687.73	714
4	5	512	9573.22	319.11	1557.20	1564
6	4	6	45.93	1.53	11.07	14
6	4	12	48.94	1.63	36.70	47
6	4	24	54.50	1.82	105.50	134
6	4	64	87.97	2.93	389.60	473
6	4	128	210.40	7.01	867.53	951
6	4	256	1158.03	38.60	1709.43	1837
6	4	512	12423.79	414.13	3119.70	3338

3	4	6	8.08	0.27	6.90	9
3	4	12	8.22	0.27	17.37	23
3	4	24	8.62	0.29	39.57	48
3	4	64	15.55	0.52	144.00	144
5	4	6	21.17	0.71	9.53	13
5	4	12	21.51	0.72	30.17	39
5	4	24	22.83	0.76	79.40	99
5	4	64	37.12	1.24	251.93	287
5	4	128	107.32	3.58	507.47	547
5	4	256	765.87	25.53	937.60	990