EFFECTIVE DESIGN AND OPERATION OF SUPPLY CHAINS FOR REMNANT INVENTORY SYSTEMS

by

Zhouyan Wang

BS, Shanghai Jiaotong University, Shanghai, P. R. China, 1997

MBA, Shanghai Jiaotong University, Shanghai, P. R. China, 2001

MS, University of Pittsburgh, Pittsburgh, PA, 2003

Submitted to the Graduate Faculty of

the School of Engineering in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2006

UNIVERSITY OF PITTSBURGH

SCHOOL OF ENGINEERING

This dissertation was presented

by

Zhouyan Wang

It was defended on

February 9th 2006

and approved by

EFFECTIVE DESIGN AND OPERATION OF SUPPLY CHAINS FOR REMNANT INVENTORY SYSTEMS

Zhouyan Wang, PhD

University of Pittsburgh, 2006

This research considers a stochastic supply chain problem that (a) has applications in a number of continuous production industries, and (b) integrates elements of several classical operations research problems, including the cutting stock problem, inventory management, facility location, and distribution. The research also uses techniques such as stochastic programming and Benders' decomposition. We consider an environment in which a company has geographically dispersed distribution points where it can stock standard sizes of a product from its plants. In the most general problem, we are given a set of candidate distribution centers with different fixed costs at the different locations, and we may choose not to operate facilities at one or more of these locations. We assume that the customer demand for smaller sizes comes from other geographically distributed points on a continuing basis and this demand is stochastic in nature and is modeled by a Poisson process. Furthermore, we address a sustainable manufacturing environment where the trim is not considered waste, but rather, gets recycled and thus has an inherent value associated with it. Most importantly, the problem is not a static one where a one-time decision has to be made. Rather, decisions are made on a continuing basis, and decisions made at one point in time have a significant impact on those made at later points. An example of where this problem would arise is a steel or aluminum company that produces product in rolls of standard widths. The decision maker must decide which facilities to open, to find long-run replenishment rates for standard sizes, and to develop long-run policies for cutting these into smaller pieces so as to satisfy customer demand. The cutting stock, facility-location, and transportation problems reside at the heart of the research, and all these are integrated into the framework of a supply chain. We can see that, (1) a decision made at some point in time affects the ability to satisfy demand at a later point, and (2) that there might be multiple ways to satisfy demand. The situation is further complicated by the fact that customer demand is stochastic and that this demand could be potentially satisfied by more than one distribution center. Given this background, this research examines broad alternatives for how the company's supply chain should be designed and operated in order to remain competitive with smaller and more nimble companies. The research develops a LP formulation, a mixed-integer programming formulation, and a stochastic programming formulation to model different aspects of the problem. We present new solution methodologies based on Benders' decomposition and the L-shaped method to solve the NP-hard mixed-integer problem and the stochastic problem respectively. Results from duality will be used to develop shadow prices for the units in stock, and these in turn will be used to develop a policy to help make decisions on an ongoing basis. We investigate the theoretical underpinnings of the models, develop new, sophisticated computational methods and interesting properties of its solution, build a simulation model to compare the policies developed with other ones commonly in use, and conduct computational studies to compare the performance of new methods with their corresponding existing methods.

Keywords: Supply Chain, Inventory Management, Production, Distribution, Facility Location, Integer Programming, Benders' Decomposition, Stochastic Programming, L-shaped Method.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

PREFACE

I first would like to appreciate the support from the National Science Foundation via Grant DMI 0217190.

I am especially indebted to my dissertation chairs, Dr. Andrew J. Schaefer and Dr. Jayant Rajgopal, for their continuous guidance, support, and inputs throughout my Ph.D program. I could not have written this dissertation without their strong encouragement and determination. Their solid academic backgrounds, invaluable comments, and kind personalities have been my life long assets and were essentially helpful in shaping the foundation of my dissertation.

I would like to express my sincere thanks to my committee members, Dr. Matthew D. Bailey and Dr. Brady Hunsaker from the Industrial Engineering department, and Dr. Prakash Mirchandani from the Operations, Decision Sciences, and Artificial Intelligence department. They contributed their valuable time and efforts to review earlier drafts of my dissertation and answered numerous questions I had. Their comments were really prompt and insightful.

My special thanks extend to all Industrial Engineering faculty, especially Dr. Bopaya Bidanda, Dr. Mainak Mazumdar, and Dr. MingEn Wang for teaching me knowledge and skills to do research and to teach.

I wish to thank all lovely Ph.D students, especially Nan Kong, Jennifer Kreke, Steven Shechter, and Lizhi Wang for many inspirational discussions.

My dissertation is dedicated to my dearest wife, Huajing Chen. Without her solid support, strong encouragement, and patient love, it would have been impossible for me to complete my degree successfully. My wonderful and most understanding parents, my father Shuling Wang and my mother Xueying Chen, receive my deepest appreciation for their generous, unconditional, and endless love towards me. My elder brother, Zhoulu Wang, has been a great supporter of my work and has accompanied me going through many tears and laughters during my studies.

1.0 INTRODUCTION

1.1 MOTIVATION

We consider a supply chain problem that integrates elements of several classical optimization problems, including the cutting stock problem, facility-location, inventory management, and distribution. Relatively large pieces, called *raws*, are manufactured or bought from suppliers and stocked at distribution centers. Demand from geographically dispersed customers arises for a variety of different lengths that must be cut at these distribution centers from the standard-sized rolls and then shipped to customers. The overall objective is to find which facilities to operate, long-run rates of replenishment for the standard sizes at each facility, and long-run policies for cutting these raws and remnants into smaller pieces to satisfy stochastic customer demands.

A large multinational steel company presented the problem addressed here. Steel is typically produced as rolls having one of several standard lengths. These rolls are then sent out to distribution centers that might either be attached to plants or at independent locations. Demands arise for a variety of different lengths that must be cut from the standard-sized rolls and shipped to customers on a daily basis. These demands are stochastic in nature and are modeled by a Poisson process. The decision of which rolls to cut is complicated by the fact that the distribution centers also maintain inventories of remnants generated from prior cutting operations, unless of course, these remnants are considered too small to satisfy any potential demand, in which case they are sold as scrap in the open market. Consequently, it is clear that a decision made at some point in time affects both the ability to satisfy demand at a later point in time, as well as the associated cost when the decision can be made in more than one way. The situation is further complicated by the fact that customer demand is stochastic and that this demand could be potentially satisfied by more than one distribution center. Given this environment, the company was interested in looking at broad alternatives for how its supply chain should be redesigned and operated in order to remain competitive with smaller and more flexible companies.

1.2 LITERATURE REVIEW

At the heart of this research is the production-distribution and facility-location problem for remnant inventory systems. The cutting stock problem is familiar to most researchers in Operations Research, and hundreds of papers (e.g., [21, 36, 37, 49, 50]) have been written on the subject and its variants since the seminal work of Gilmore and Gomory [41, 42, 43]. The interested reader is referred to Cheng et al. [23], or Haessler and Sweeney [46] as starting points on the aforementioned topic. The basic cutting stock problem is typically a static one [53, 68, 69, 90]. Both optimal as well as heuristic methods have been applied to solve the problem [97, 103, 104, 105, 110]: the product is produced in (a relatively small number of) standard sizes, and one must meet a given set of requirements for this product. These requirements encompass (a relatively large number of) non-standard sizes that must be cut from the standard stock. The problem is to find the number of pieces of stock of each standard size to use, along with the associated cutting patterns, so that all requirements are satisfied at minimum cost [23]. This situation is very common in a number of continuous production applications including metals such as steel or aluminum, paper, textiles, lumber, fiber-optic and electrical cables, and glass.

The cutting stock problem presented in this research is a little different. First, we address the problem in the context of a supply chain where a company has geographically dispersed distribution centers that can be stocked with standard sizes that are procured from its plants; the demand at these centers for smaller sizes comes from other geographically distributed points. Second, unlike classical cutting stock problems, in our model decisions are made in a *stochastic* and *dynamic* environment. Third, we account for a sustainable manufacturing environment where the trim is not necessarily treated as waste; rather, it may get recycled. This assumption holds in industries such as steel, or aluminum. Fourth, and most importantly, the problem is not static, where a one-time decision has to be made. Rather, decisions are made on an ongoing basis since demand is dynamic, and decisions made at one point in time have a significant impact on those made later.

Steel can either be produced in integrated mills from iron ore, or in so-called mini-mills from recycled scrap. Much of the relevant research has focused on optimizing steel production planning and scheduling [78, 88, 99, 115]. A survey of this topic may be found in Tang et al. [98], where numerous approaches for solving scheduling and planning problem in the steel industry are reviewed. A few papers have looked at the combined production and distribution system, e.g., Chen and Wang [22] present a deterministic linear programming model for this integrated production and transportation planning problem. Inventory planning in the steel industry is addressed by Denton et al. [35], who present a model of a hybrid make-to-stock, make-to-order system to help an integrated steel mill manage its inventory. Krichagina et al. [61] use a combination of linear programming and Brownian analysis to address a problem in a dynamic and stochastic environment, but this paper looks at a single location, and only considers the production problem. Denton and Gupta [34] describe a model where they use a two-stage stochastic integer programming model to decide on optimal levels of semi-finished steel inventory, for use with a delayed product differentiation approach.

In addition to steel, researchers have also addressed production and inventory issues in other remnant inventory systems such as aluminum, copper, and paper and pulp. For example, Hendry et al. [47] consider a two-stage production planning problem in the copper industry and describe an IP-based heuristic method to solve the problem; Johnson et al. [57] address a combined skiving and cutting stock problem in paper mills; Bredström et al. [18] develop MIP models for the production and distribution problem in pulp mills and provide a heuristic based on column generation; and Partington [82] addresses inventory issues related to aluminum.

The classical transportation problem first described by Hitchcock in 1941 [51]. In the area of integrated production and transportation, Hochbaum et al. [52] show that for the production-distribution problem, it is still NP-hard even if demand is deterministic and the production cost function is concave, separable and symmetric. While some authors [31, 56, 81, 91, 109] focus on strategic models that integrate design decisions such as location, plant capacity, and transportation channels into the supply chain, our research is involved more with both production and distribution operations. In this area, some authors [16, 17, 85, 113] consider decisions across multiple stages, and others [10, 14, 15, 20, 95] address problems that integrate inventory and distribution decisions.

For algorithms for this NP-hard production-distribution problem, the paper by Tuy et al. [101] is a good starting reference since they look into a problem with two production facilities. Tuy et al. [102] develop a polynomial time algorithm for the simple case of two production facilities with concave production costs and linear transportation costs. Kuno and Utsunomiya [62] present a Lagrangian-based branch-and-bound algorithm for the situation where the production cost function is concave and separable. All of these papers only consider deterministic demand.

The classical network location problems include the maximum covering location problem [32, 79], the set covering location problem [9, 30, 100], the p-center problem [19, 70, 83, 87], and the facility location problem [5, 8, 27, 66], which includes the uncapacitated version (UFLP [4, 44]) as well as capacitated version (CFLP [7, 40, 60]). In general, both UFLP and CFLP are NP-hard [77] and have been widely studied. Many algorithms and heuristics have been developed in the past decades [7, 38, 55, 59, 106]. An exhaustive survey can be found in [6], while a more recent review of algorithms, heuristics, and exact methods for the capacitated facility location problem may be found in [96].

Some research has also been conducted on the polyhedral structure of the NP-hard facility location problem. For the capacitated facility location problem, Leung and Magnanti [65] look into the polyhedral structure for the case of constant, equal capacities; Aardal et al. [1] study the case of different capacities; Deng and Simchi-Levi [33] consider the case of unsplit demands; Cornuéjols at al. [28] compare the strength of various Lagrangian relaxations. Interested readers are also referred to polyhedral study for the uncapacitated facility location problem [24, 25, 29, 45].

The problem most closely related to ours is the multi-product capacitated facility location problem (MPCFL), first introduced by Lee [63, 64]. Relatively few researchers have studied this problem. Mazzola and Neebe [75] show that it is NP-hard and presented a solution method based on Lagrangian relaxation method.

In this research, we extend the deterministic location-production-distribution problem to the stochastic case. Stochastic programming has been well studied, e.g. Birge and Louveaux [11], Higle and Sen [48], and Kall and Wallace [58]. For our problem in this case, the decision is further complicated by the fact that there exist several scenarios in which some parameters, i.e. the cutting costs, raw costs, demand rates, etc., will have different values under different scenarios. Specifically, we consider facilities that are capacitated with the types, sizes, and costs of raw materials processed, cutting costs, capacities, and scrap value possibly being different under different scenarios. The types, sizes, and amounts demanded by customers are also stochastic variables. Finally, the shipping costs from the facilities to the customers can change over scenarios as well.

In examining prior research, the most closely related to this research is some relatively recent work (Adelman and Nemhauser [2]; Adelman et al. [3]) applied to the fiber-optic cable industry. In these papers, the authors consider a single supply and a single demand location and build a linear programming model that allows standard duality results to place a "value" on remnants of any size; this in turn leads to a cutting policy adapted to a dynamic demand environment wherein one chooses the size of the piece to be cut based upon the reduction in its value when it is cut to a remnant of smaller size.

Our work extends the work in Adelman and Nemhauser [2] in several regards. First, unlike the fiber optic cable industry, scrap in the metals industry has value, and our models account for this. For example, used beverage can scrap value could be as high as 75% of the original value [80], and coil steel scrap value could be more than 25% of the original value [86, 92]. Second, we do not assume prespecified proportions in which different raw stock sizes are replenished, but rather, allow our model to find the optimal replenishment rates for each location; correspondingly, the model is far more realistic in practice. Third, the decision maker in our model incurs a cost for each cutting operation. The cutting cost is nontrivial in the steel industry (Chu and Antonio [26]) and ignoring this cost is inaccurate in real-life practice. Fourth, unlike Adelman and Nemhauser [2] who consider a single location, we consider multiple supply and demand locations. The modified model is then embedded within a location/distribution network for the general supply chain problem in order to determine which distribution facilities should be selected for operation, what cutting policies should be used, and how product should be distributed from distribution centers to demand points. Fifth, we consider the fixed cost and the capacity limit for each facility. Rather than opening all facilities, we also let the model itself choose the optimal set of facilities to open from the potential locations. Sixth, unlike Adelman and Nemhauser [2] who consider the deterministic problem, we make the model even more realistic by introducing stochastic parameters.

1.3 SIGNIFICANCE

The research is significant in several regards. First, it represents one of the few efforts at an exact solution to a supply chain problem that integrates location, production and distribution. Second, the application domain is potentially vast since many industries, such as paper, steel and aluminum, deal with remnant inventories similar to those addressed here. Third, the research aims to improve solution procedures for large-scale stochastic programming problems, which is an area that has been relatively unexplored because of the computational difficulties associated with it.

In this dissertation, major contributions are as follows. First, to effectively manage long-run costs in a dynamic and stochastic demand environment, we develop a policy to assist price-directed policy makers in deciding which facility to satisfy the demand from and to pick which remnant to cut to obtain the demanded size. Second, to make the policy applicable, we develop a new perturbation technique to make the dual prices satisfy several desirable properties. Third, considering the facility location problem results in an NP-hard mixed-integer program. To solve the resulting large-scale integer program, we adapt Benders' decomposition [74]. To improve its performance, we develop a new method to generate a set of feasibility constraints that replace Benders' feasibility cuts and whose number is bounded polynomially. Fourth, by considering stochastic demand, we develop a stochastic program and present new algorithms based on the L-shaped method to solve the resulting problem. Fifth, it also shows the potential of applying new feasibility constraint in complex solution methodologies because our results prove new feasibility constraints are robust in complex solution methodologies such as Benders' Decomposition and the L-shaped method. Finally, our solution development provides insight into algorithmic and computational issues regarding these methodologies.

This document is organized as follows. We first start in Chapter 2 with a linear programming model for the actual cutting and distribution problem, given a set of locations, and then present a strategy for dynamic demands. In Section 2.5, we address inventory control and present a simulation model to evaluate the technique that is developed. In Chapter 3, the LP model is then extended to the case where the location of distribution centers must be decided by considering their different fixed costs and capacity limits. Solution algorithms based on Benders' decomposition are proposed for this problem. In Chapter 4, we present an even more general model with stochastic parameters, and develop a stochastic programming approach based on the L-shaped method to solve this problem. Chapter 5 presents conclusions and directions for future research.

2.0 BASIC MODEL: PRODUCTION AND DISTRIBUTION

In our first model, we assume that all facilities are open and have no capacity limits. Each facility stocks some set of standard raw sizes. Demand arises from the various customers according to a Poisson process with different mean values. When a certain demand size arises, an effective policy should help the decision makers choose an appropriate raw or remnant at one of the facilities so that material, production, and transportation costs are minimized in the long run. The objective of this model is to find such a policy.

2.1 SYSTEM DESCRIPTION

As shown in Figure 1, we consider a production-distribution system consisting of a set of geographically distributed facility locations indexed by $i \in I$ each of which stocks a product in one or more "standard" sizes. The product is replenished periodically from an outside production facility. Demand for the product arises from some other set of geographically distributed demand locations indexed by $j \in J$, but typically in smaller, "non-standard" sizes. The demand for a particular size k at a particular location j is assumed to follow a Poisson distribution with some known mean rate λ_{jk}. We assume that there is no capacity limit at each facility i and that all demand must be fulfilled. A cost δ_i is incurred each time a cut is made at location i and a per-unit transportation cost c_{ijk} is incurred in shipping a unit of size k from location i to location j. The problem is to determine a long-term policy for cutting and distributing the product in order to satisfy demand.

In order to formulate the problem first note that remnants from cutting standard-size stock can either be used again to produce other (smaller) non-standard sizes, or recycled

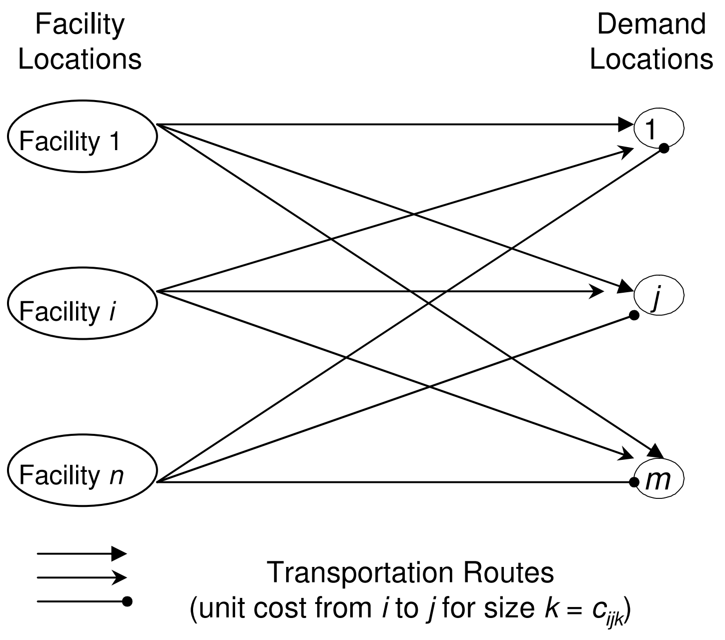

Figure 1: Distribution Network

as scrap. Thus, at any given time each distribution facility could be expected to have an inventory of units of various sizes. Now consider a node in Figure 1 that represents some distribution facility i. Each such node will be represented by its own sub-network. These sub-networks are then connected to the various demand locations as part of a larger distribution network; thus the overall network is one comprising of several smaller sub-networks. Each of the sub-networks is represented using a model similar to that proposed in [2] for their single supplier/single user problem. In this model, nodes in each sub-network represent the various sizes that could be present at the facility at any time as standard (raw) stock, finished product, remnants or scrap, where node m represents size m. The flow along an arc from node m to node n (where $n < m$) represents the number of units of size m that are cut down to remnants of size n (while producing that many pieces of size $m - n$ in the process); Figure 2 illustrates an example of such a sub-network.

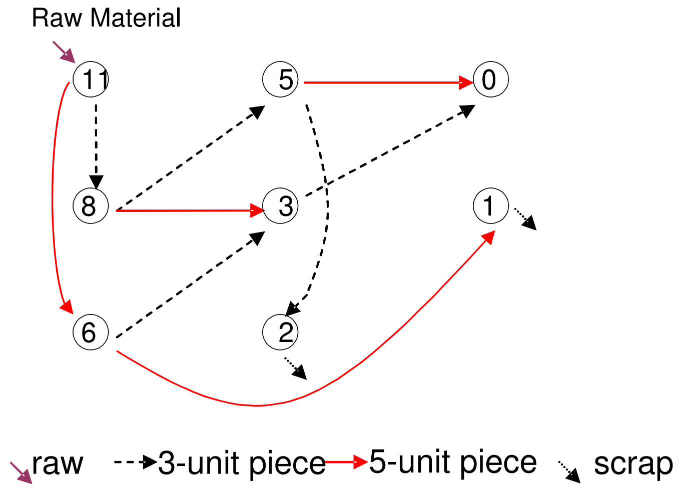

Figure 2: Sub-Network

2.2 MODEL FORMULATION

To describe our model, we first introduce some notation.

- I = Set of facility locations.
- J = Set of demand locations.
- K_i = Set of all possible sizes, including scrap sizes (we refer to sizes that are smaller than the smallest demanded size as scrap sizes), that could be generated at facility i (we include the size 0 in this set); define $K = \bigcup_{i \in I} K_i$.
- D_j = Set of all sizes that are demanded at demand location $j \in J$; define $D = \bigcup_{j \in J} D_j$.
- $S_i \subseteq K_i$ = Set of all raw stock sizes that are processed at facility location $i \in I$; define $S = \bigcup_{i \in I} S_i$.
- λ_{jk} = Mean demand rate at location $j \in J$ for product of size $k \in D_j$.
- c_{ijk} = Cost of transporting a single unit of size $k \in D_j$ from location $i \in I$ to location $j \in J$.
- a_{ih} = Cost of one unit of raw stock of size $h \in S_i$ at location $i \in I$.

- σ_i = Unit salvage value of product at facility $i \in I$; $\sigma_i > 0$.
- δ_i = Cost per cut at facility $i \in I$; $\delta_i > 0$.

The variables are:

- x_{ijk} = Rate of shipment of units of size $k \in D_j$ from facility location $i \in I$ to demand location $j \in J$.
- $y^i_{m,n}$ = Rate of generation at facility location $i \in I$, of units of size $(m-n)$ that are obtained by cutting size $m(\geq n)$ down to size n.
- r_{ih} = Rate of replenishment of raw units of size $h \in S_i$ at facility location $i \in I$; define $r_{ih} = 0$ if $h \in (K_i \backslash S_i)$.
- s_{iu} = Rate at which size u is scrapped at facility location $i \in I$. Define $s_{iu} = 0$ if the size u cannot be generated at location i.

Before formulating the problem, we also define for each $i \in I$, the sub-network $G_i = (K_i, A_i)$ similar to the example in Figure 2, where K_i is the set of nodes and A_i the set of arcs in the graph. Specifically, we define:

$$A_i = \{(m,n) | (m-n) \in D; m, n \in K_i, m > n\}.$$

Note that for the sub-network in Figure 2, $K_i = \{0, 1, 2, 3, 5, 6, 8, 11\}$ and the arc set $A_i = \{(11,8),(11,6), (8,5),(8,3),(6,3),(6,1),(5,2),(5,0),(3,0)\}$.

We are now ready to formulate the problem as the following linear program (LP):

$$\text{Minimize} \sum_{i \in I} \sum_{h \in S_i} a_{ih} r_{ih} + \sum_{i \in I} \sum_{\{(m,n) \in A_i | n>0\}} \delta_i y^i_{m,n} + \sum_{i \in I} \sum_{j \in J} \sum_{k \in D_j} c_{ijk} x_{ijk} - \sum_{i \in I} \sum_{u \in K_i} u \sigma_i s_{iu} \quad (2.1)$$

subject to:

$$\left(r_{il} + \sum_{m|(m,l) \in A_i} y^i_{m,l} \right) - \left(s_{il} + \sum_{n|(l,n) \in A_i} y^i_{l,n} \right) = 0, \text{for all } i \in I \text{ and } l \in K_i, \quad (2.2)$$

$$\sum_{i\in I} x_{ijk} = \lambda_{jk}, \text{ for all } j \in J \text{ and } \ k \in D_j, \tag{2.3}$$

$$\left(\sum_{(m,n)\in A_i|(m-n)=k} y^i_{m,n}\right) - \left(\sum_{j\in J} x_{ijk}\right) = 0, \text{ for all } i \in I \text{ and } k \in D, \tag{2.4}$$

$$\text{all } x, y, r, s \geq 0. \tag{2.5}$$

It is readily seen that the objective (2.1) is to minimize cost of raw stock, cutting and transportation, less the value of the scrap that is salvaged. Constraints (2.2) ensure flow balance at each node of each facility sub-network, constraints (2.3) ensure that all demand is satisfied, and constraints (2.4) ensure that total flow of each size out of each sub-network is equal to its production in that sub-network. Notice that this primal formulation only yields a replenishment and shipment policy for a static environment for one particular demand rate (the expected value λ_{jk}). What one actually needs is a policy to use in a dynamic environment where demand occurs in a stochastic fashion. Furthermore, additional difficulty arises from the fact that in general the above LP will have multiple optima.

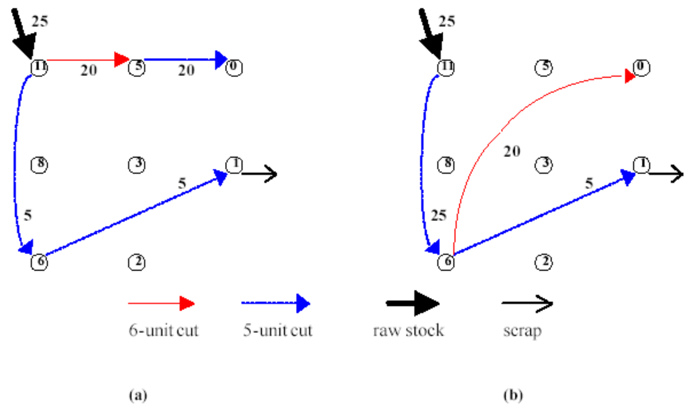

Figure 3: Alternative Optima

For instance, consider Figure 3: If the flow in the Figure 3(a) is optimal, then so is the one in Figure 3(b). Both take in 25 units of raw stock of size 11 and generate 30 units of size

5 and 20 units of size 6, along with 5 units of scrap of size 1. While there is no difference between one optimum and the other for the static problem formulated in this section, the same cannot be said if the objective is to come up with a long-term operating policy for the case where the demands are dynamic. In such an environment a particular decision could have a significantly more (or less) adverse impact on future decisions.

In light of the preceding discussion we introduce the dual program corresponding to the above LP. Define the dual variables:

- $\eta_{i,m}$ corresponding to each constraint in (2.2), i.e., to each node of each facility's sub-network,
- μ_{jk} corresponding to each constraint in (2.3), i.e., to each demanded size at each location,
- π_{ik} corresponding to each constraint in (2.4), i.e., to each demanded size produced at each location.

Then the corresponding dual is:

$$\text{Maximize} \quad \sum_{j \in J} \sum_{k \in D_j} \lambda_{jk} \mu_{jk}, \tag{2.6}$$

subject to:

$$\mu_{jk} - \pi_{ik} \leq c_{ijk}, \text{ for all } i \in I, j \in J, k \in D_j, \tag{2.7}$$

$$\eta_{i,n} - \eta_{i,m} + \pi_{ik} \leq \delta_i, \text{ for all } i \in I; m, n \in K_i; (m,n) \in A_i; k = m - n, n > 0, \tag{2.8}$$

$$\eta_{i,n} - \eta_{i,m} + \pi_{ik} \leq 0, \text{ for all } i \in I; m, n \in K_i; (m,n) \in A_i; k = m - n, n = 0, \tag{2.9}$$

$$\eta_{i,h} \leq a_{ih}, \text{ for all } i \in I, h \in S_i, \tag{2.10}$$

$$\eta_{i,u} \geq u\sigma_i, \text{ for all } i \in I, u \in K_i, \tag{2.11}$$

$$\eta, \mu, \pi \text{ unrestricted.}$$

In order to interpret the dual, consider an external entity that is willing to deliver the demanded pieces directly to the customer at the demand point. Then μ_{jk} may be interpreted as the sale price set by this entity for a unit of size k at demand point j. The dual objective is to maximize revenues. The same entity is also willing to supply pieces at distribution facility i for the price of π_{ik} for a piece of size k. We may interpret $\eta_{i,m}$ as the inherent unit value of a product of size m at facility i. Note that the $\eta_{i,m}$ values provide us with the value associated with a particular size of remnant. For our long-term decision-making, these are the values that will be used.

Then (2.7) states that the sale price at location j can be no higher than the price charged at facility i plus the shipping cost from facility i to demand location j (otherwise the decision maker would buy at the facility at the price of π_{ik} and ship it himself). To interpret constraints (2.8) and (2.9) first note that $\eta_{i,m}$ - $\eta_{i,n}$ is the loss in inherent value when a piece of size m is cut down to a piece of size n (while producing a piece of size $k{=}(m-n)$ in the process). Thus the quantity $\eta_{i,m}$ - $\eta_{i,n} + \delta_i$ may be interpreted as the "total cost" to us of making a piece of this size k. Then (2.8) and (2.9) state that the price charged at facility i for size k cannot exceed the cost to make it (otherwise the decision maker would make it himself and not buy from the external entity); note that in (2.9) we do not include cutting costs since the entire remnant is used and there is no cutting operation. Constraints (2.10) and (2.11) are defined for each facility; (2.10) says that a unit of size k cannot be worth more than its raw unit cost; and (2.11) says that a unit of size k is worth at least as much as its scrap value.

2.3 A PRICE-DIRECTED STRATEGY FOR DYNAMIC SUPPLY CHAIN MANAGEMENT

2.3.1 Strategy in a Dynamic Demand Environment

In the previous section we have presented primal-dual formulations for a *static* version of our problem, i.e., we have assumed a fixed value for the demand rate for each size at each location. In practice, demand is stochastic and is modeled by a Poisson process with some rate vector λ. Thus, what one needs is a *policy* that can be used to make demand fulfillment decisions in this dynamic demand environment. To do this, we examine the optimal solution to the static problem and use it to arrive at a sensible policy for the dynamic problem. The dual program just described is crucial to this step since the policy will be based on the values of the optimal dual variables.

As an example of what this entails, suppose that a customer at demand location j orders a unit of size 5. There may be multiple facilities from which this demand could be satisfied as well as multiple cutting options at a particular facility. Since a decision made today will influence the cost of a decision made tomorrow, the questions to be answered are of the following kind: "Should we cut this order from our 11-unit inventory at Chicago, from our 13-unit inventory at Chicago, or from our 11-unit inventory at Pittsburgh?" Recall that $\eta_{i,m}$ is the dual variable corresponding to constraint (2.2) for a remnant of size m at location i. Thus the optimal (static) value for $\eta_{i,m}$ represents the marginal value of an extra piece of size m at location i, i.e., the inherent value of this particular remnant. There is intuitive appeal in using this value as the basis for a policy that is adapted to the dynamic demand environment. In particular, the answer to the question posed above will depend on (a) the reduction in inherent value when going from a larger to a smaller size (e.g., 13 to 8 units or 11 to 6 units, as represented by $\eta_{i,13} - \eta_{i,8}$ or $\eta_{i,11} - \eta_{i,6}$, respectively), (b) the cutting costs δ_i at Pittsburgh and Chicago, and (c) the transportation cost from the source to the destination.

The fundamental question to answer is the following: "What is the general policy that we should follow for choosing from multiple cutting and distribution options in order to minimize long-run costs?" Obtaining a dual solution to the static problem allows us to develop such a policy for the dynamic case and also provides us with information as to when we might be indifferent between one policy and another.

Theorem 1. *Suppose there is demand at location j for units of size k (that is, $k \in D_j$ and $\lambda_{jk} > 0$). The optimal static strategy for fulfilling this is to choose the location i' and cutting size m' via: $(i', m') \in argmin_{i \in I, (m \in K_i | m \geq k)} (\eta_{i,m} - \eta_{i,m-k} + \delta_i + c_{ijk})$, where $\delta_i = 0$ if $m = k$.*

Proof. Consider the set $I_k = \{i \in I | \exists m \geq k, y^i_{m,m-k} > 0\}$ at optimality. That is, I_k is the set of all facilities that produce units of size k according to the optimum solution. We would like to see which of these facilities could possibly have $x_{ijk} > 0$ at optimality, i.e., be used to fulfill the demand arising at location j for size k. Suppose then that $i' \in I_k$ is such a facility, i.e., $x_{i'jk} > 0$ at optimality.

From constraint (2.7) of the dual problem we know that $\mu_{jk} \leq \pi_{ik} + c_{ijk}$ for all $i \in I$. Since (a) μ_{jk} appears only in (2.7), (b) $\lambda_{jk} \geq 0$, and (c) the dual objective is to be maximized, it also follows that

$$\mu_{jk} = \min_{i \in I}(\pi_{ik} + c_{ijk}). \tag{2.12}$$

Now consider complementary slackness conditions corresponding to (2.7) and (2.8):

$$(\mu_{jk} - \pi_{ik} - c_{ijk})x_{ijk} = 0, \text{ for all } i \in I, j \in J, k \in D_j \tag{2.13}$$

$$(\eta_{i,n} - \eta_{i,m} - \delta_i \ + \pi_{ik})y^i_{m,n} = 0, \text{ for all } i \in I, (m,n) \in A_i, k = m - n \tag{2.14}$$

Since $x_{i'jk} > 0$, complementary slackness condition (2.13) implies $\mu_{jk} = \pi_{i'k} + c_{i'jk}$, so that from (2.12), $i' \in argmin_{i \in I}(\pi_{ik} + c_{ijk})$. Also, from complementary slackness condition (2.14), for all $i \in I_k$, if size m' is cut down to size $n = (m' - k)$, (so that $y^i_{m',n} > 0$), it follows that $\pi_{ik} = \eta_{i,m'} - \eta_{i,(m'-k)} + \delta_i$. Since all defined $c_{ijk} > 0$ it follows from the two previous sentences that $(i', m') \in argmin_{i \in I,(m \in K_i | m \geq k)}(\eta_{i,m} - \eta_{i,(m-k)} + \delta_i + c_{ijk})$. □

Theorem 1 implies that if there is demand for size k, the optimal strategy in the static problem is to consider all pieces of size $m \geq k$ at all facilities and pick a facility and size for which $(\eta_{i,m} - \eta_{i,m-k} + \delta_i + c_{ijk})$ or $(\eta_{i,m} - \eta_{i,0} + c_{ijk})$ is minimized, depending on whether $m > k$ or $m = k$ respectively. The intuition behind this is straightforward: the difference $\eta_{i,m} - \eta_{i,m-k}$ is the decrease in value to the decision-maker associated with reducing a unit of size m to size $m - k$ at facility location i and, along with δ_i, thus represents a "cost." The additional term c_{ijk} is added to account for the transportation cost from facility i to demand location j. Furthermore, since we assume no limits on capacity, all of the demand from customer j for size k would be satisfied from the same facility i.

We use Theorem 1 as the basis for our policy in a dynamic demand environment by restricting our search to the existing inventory at each facility. The optimal size corresponding

Table 1: Facility Data

Facility 1				Facility 2			
Raw Size	Cost	Unit Scrap Value	Cutting Cost	Raw Size	Cost	Unit Scrap Value	Cutting Cost
13	20	0.05	0.05	11	19	0.1	0.05
11	19						

to a static policy might not be in stock at the associated facility and there is thus no guarantee that the optimal static policy will always be used in a dynamic demand environment. However, it does provide us with a logical basis for a policy in the dynamic environment. Such policies based on the inherent values of inventory are referred to as price-directed policies. We illustrate this policy by a small numerical example in the next section.

2.3.2 A Numerical Example

In this section, we provide a small numerical example to show how the policy works. Consider a system with two facilities and two demand locations. The raw sizes available at each facility along with the unit costs are summarized in Table 1. This table also provides the unit salvage values at each facility. The demanded sizes and the corresponding demand rates at each demand point, along with unit shipping costs are summarized in Table 2.

Upon formulating and solving this problem we obtain the dual prices given in Table 3. Note that there are certain sizes that can never be produced as remnants, and the $\eta_{i,k}$ are not defined for these.

To see how the solution translates into a dynamic policy, consider a case where there is demand for 3-unit pieces at Location 2. Suppose that at the time that this demand arrives, we have 13-unit pieces and 6-unit remnants at Facility 1 and 6-unit remnants at Facility 2. In this case, facility 1 has: $\eta_{1,13}-\eta_{1,10}+c_{123}+\delta_1 = 20-17+2+0.05 = 5.05, \eta_{1,6}-\eta_{1,3}+c_{123}+\delta_1 =$

Table 2: Unit Shipping Costs and Demands

	Location 1		Location 2	
Size	5	6	3	5
Demand	20	20	10	30
Facility 1	2	4	2	5
Facility 2	3	3	1.5	4

Table 3: Optimal Dual Prices (η_{ik})

Remnant Size	Facility	
k	$i = 1$	i =2
13	20	-
11	19	19
10	17	-
8	11.5	15.5
7	8.6	-
6	8.55	9.55
5	8.50	9.45
4	3.05	-
3	3	3.5
2	0.1	0.2
1	.05	0.1

$8.55-3+2+0.05 = 7.60$ and facility 2 has: $\eta_{2,6}-\eta_{2,3}+c_{223}+\delta_2 = 9.55-3.5+1.5+0.05 = 7.60$. We would thus use the 13-unit piece at Facility 1 to fulfill this demand. On the other hand if we only had 6-unit remnants at both locations, we would be indifferent between the two. While this is a small example, it nevertheless illustrates the kinds of policies that we can derive from the duality results.

2.3.3 Degeneracy, Multiple Optima, and Perturbation

In Section 2.2 at Chapter 2, we noted with an illustrative example that the primal problem will generally have multiple optima. When attempting to adapt the static optimum to a dynamic policy, an issue that needs to be addressed is that the problem tends to be degenerate at optimality, with several of the $y^i_{m,n}$ variables being basic at zero values. Each of the optimal bases leads to a different dual vector. Note that this should be intuitively obvious based on the fact that most dual variables have objective coefficient of zero. Unfortunately, not all of these dual vectors yield values for the optimal dual prices that are desirable from the perspective of price-directed policies. There is no guarantee that an arbitrary dual vector will satisfy properties that are intuitively appealing with the interpretation of the dual price for a remnant as its inherent value. Adelman and Nemhauser [2] present several such properties in their work on the single location problem with no cutting costs and no scrap values. We present monotonicity and superadditivity as follows:

Definition 1. *A value function η is* monotonic *if for any facility $i \in I$ and any sizes $m, m-n \in K_i$, $\eta_{i,m} \geq \eta_{i,m-n}$.*

Definition 2. *A value function η is* superadditive *if for any facility $i \in I$ and any sizes $m, m-n, n \in K_i$, $\eta_{i,m} \geq \eta_{i,m-n} + \eta_{i,n}$.*

We now generalize the above concepts as follows:

Definition 3. *For a given cost $\delta \in \Re^{|I|}_{++}$, a value function η is* δ-monotonic *if for any facility $i \in I$ and any sizes $m, m-n \in K_i$, $\eta_{i,m} \geq \eta_{i,m-n} - \delta_i$.*

Definition 4. *For a given cost $\delta \in \Re^{|I|}_{++}$, a value function η is* δ-superadditive *if for any facility $i \in I$ and any sizes $m, m-n, n \in K_i$, $\eta_{i,m} \geq \eta_{i,m-n} + \eta_{i,n} - \delta_i$.*

Consider any piece that has been cut into two smaller pieces. If a value function is δ-*monotonic*, then the inherent value of this piece plus the unit cutting cost is at least as much as the inherent value of any of the smaller pieces. If a value function is δ-*superadditive*, then the inherent value of the piece plus the unit cutting cost is at least as much as the sum of the inherent values of the two smaller pieces.

When $\delta = 0$ for all facilities, the notions of δ-monotonicity and δ-superadditivity reduce to classical notions of monotonicity and superadditivity. The interested reader is referred to Adelman and Nemhauser [2] for a further discussion of these types of properties when $\delta = 0$ for a single facility problem. It suffices to say that such properties should be satisfied by the value function if we plan to use them for a dynamic policy, but picking an arbitrary optimal dual vector for our problem does not necessarily guarantee this.

What we need, then, is a consistent way of picking the "correct" values for the optimal dual variables for use in a dynamic environment where one must make decisions over time based on these inherent values. To address this issue we adapt the approach followed in Adelman and Nemhauser [2] of developing a so-called "ϵ-perturbation" that creates small exogenous supplies and demands to ensure positive flows in optimal arcs. Our perturbation differs from the one in Adelman and Nemhauser [2] in several respects. First, with multiple facilities we cannot simply create exogenous demand to ensure positive flows, since this demand may be completely satisfied by a single facility and still leave the flows in all the others degenerate. Rather, one needs to account for each facility individually and ensure that there is (at least some minimal amount of) production of all sizes at each facility so that we avoid $y^i_{m,n} = 0$. Second, unlike in Adelman and Nemhauser [2] we do not assume that the relative proportions of raw sizes at each facility is prespecified. Rather, these are decision variables in our problem. Finally, we do not necessarily create perturbations for every size.

To introduce the perturbation we first extend the set D, which was defined earlier as the union of all sizes demanded across all locations. We now define a set Δ by adding to D every other size that could be generated at a facility i, i.e.

$$\Delta = K \cup D. \tag{2.15}$$

We partition Δ as follows,

Definition 5. *Define*

a) $\Delta_0 = \{m | m \in \Delta, m < \min\{k | k \in D\}\}$,

b) $\Delta_1 = \{m | m \in \Delta, m \geq \min\{k | k \in D\}\}$.

Set Δ_0 indexes scrap sizes while Δ_1 indexes all sizes in the extended demand set Δ that are at least as large as the minimum demand size in the original problem.

Next suppose ϵ is some small positive constant.

Definition 6. *The parameters of an ϵ-perturbation are defined as $\alpha_{im} \in \Re^1_{++}, \beta_{im} \in \Re^1_{++}, \forall i \in I, m \in \Delta$ such that:*

$\beta_{im} < \alpha_{im}, \forall m \in \Delta_0$ *and* $\beta_{im} = \alpha_{im}, \forall m \in \Delta_1$.

To define an ϵ-perturbation, we go through the following three steps:

Step 1: Solve the primal LP problem and denote the optimal values of x_{ijk} by x^*_{ijk}.

Step 2: Extend the set D to Δ as given by (2.15), the union of the original demand sizes and all sizes that could possibly be generated. Define new node and arc sets for each i as K'_i and A'_i, respectively.

Step 3: At each facility i, perturb the right hand sides of constraint (2.2) for nodes $l \in \Delta$ as follows:

$$\left(s_{il} + \sum_{n|(l,n)\in A'_i} y^i_{l,n}\right) - \left(r_{il} + \sum_{m|(m,l)\in A'_i} y^i_{m,l}\right) = \alpha_{il}\epsilon, \text{ for all } i \in I \text{ and } l \in \Delta. \tag{2.16}$$

For nodes $l \in K'_i \backslash \Delta$, constraint (2.2) is kept the same as before:

$$\left(s_{il} + \sum_{n|(l,n)\in A'_i} y^i_{l,n}\right) - \left(r_{il} + \sum_{m|(m,l)\in A'_i} y^i_{m,l}\right) = 0, \text{ for all } i \in I \text{ and } l \in K'_i \backslash \Delta. \tag{2.17}$$

Replace x_{ijk} in constraint (2.4) with x^*_{ijk} and perturb constraint (2.4) as follows for the size $k \in D$:

$$\sum_{(m,n)\in A'_i|(m-n)=k} y^i_{m,n} = \left(\sum_{j\in J} x^*_{ijk} + \beta_{ik}\epsilon\right), \text{ for all } i \in I \text{ and } k \in D. \tag{2.18}$$

For the size $k \in \Delta \backslash D$, there is no original demand and thus, $x^*_{ijk} = 0$. Therefore, we perturb constraint (2.4) as follows:

$$\sum_{(m,n)\in A'_i|(m-n)=k} y^i_{m,n} = \beta_{ik}\epsilon, \text{ for all } i \in I \text{ and } k \in \Delta\backslash D. \tag{2.19}$$

Finally, we drop constraint (2.3) in the perturbed model.

We can rewrite the primal perturbed problem as follows:

Minimize $\sum_{i\in I}\sum_{h\in S_i} a_{ih}r_{ih} + \sum_{i\in I}\sum_{\{(m,n)\in A_i|n>0\}} \delta_i y^i_{m,n} + \sum_{i\in I}\sum_{j\in J}\sum_{k\in D_j} c_{ijk}x^*_{ijk} - \sum_{i\in I}\sum_{u\in K_i} u\sigma_i s_{iu}$

subject to (2.16), (2.17), (2.18), (2.19), and $y, r, s \geq 0$.

The dual of the perturbed problem is:

Maximize $\sum_{i\in I}\sum_{k\in\Delta} \pi_{ik}(\sum_{j\in J} x^*_{ijk} + \beta_{jk}\epsilon) - \sum_{i\in I}\sum_{m\in\Delta} \eta_{i,m}\alpha_{im}\epsilon$

Subject to (2.8), (2.9), (2.10), (2.11), and η, π unrestricted.

Even if the size $k \in K_i$ is not generated at location i for shipment to some demand point (i.e., x_{ijk} is zero for all j), the constraints in the above perturbed model still force production of a small amount ($\beta_{ik}\varepsilon$) of this size. For sizes that are generated for shipment, the production is simply increased by a small amount.

Before getting into details about what it accomplishes, we examine the perturbation further and contrast the optimal flows in the original problem with those in the perturbed problem. Note that in the perturbed problem, for each m that belongs to Δ:

- There is an extra inflow of $\alpha_{im}\epsilon$ units into node m.
- There is an additional demand of $\beta_{im}\epsilon$ units for size m, where $\beta_{im} < \alpha_{im}$ if $m \in \Delta_0$ and $\beta_{im} = \alpha_{im}$ if $m \in \Delta_1$.

We first show that in the optimal solution to the perturbed problem there is always a positive flow from all nodes with scrap sizes to node 0.

Proposition 1. *For every optimal solution* $^*y$ *to the perturbed problem,* $\forall i \in I, m \in \Delta_0$*:* $^*y_{m,0} = \beta_{im}\epsilon$*.*

In order to prove Proposition 1, we first present the following definition and lemma.

Definition 7. *Define* $P^i_{m,n}$ *as the set of directed paths* p *that start at node* m *and end at node* n *in network* $G'_i = (K'_i, A'_i)$ *at facility* $i \in I$*. Define* $^*q^i_p = \min_{(m,n)\in p} {}^*y^i_{m,n}$*, that is, the optimal path flow along* p*, and* $^*Q^i_{m,n} = \sum_{p\in P^i_{m,n}} {}^*q^i_p$*.*

Lemma 1. *Under an* ϵ*-perturbation,* $\forall i \in I, m \in K_i$*,* $^*Q^i_{m,0} \geq \beta_{im}\epsilon$*.*

Proof. Suppose at the optimum $^*Q^i_{m,0} < \beta_{im}\epsilon$, for some $i \in I, m \in K_i$. First, note that the total flow along all arcs $(m_1, m_2) \in A'_i$ such that $m_1 - m_2 = m$ must be at least as much as $\beta_{im}\epsilon$, so that

$$\sum_{(m_1,m_2)\in A'_i | m_1-m_2=m} {}^*y^i_{m_1,m_2} \geq \beta_{im}\epsilon > {}^*Q^i_{m,0}.$$

Isolating the flow along arc $(m, 0)$, this implies that

$${}^*y^i_{m,0} + \sum_{(m_1,m_2)\in A'_i | m_1-m_2=m, m_1\neq m} {}^*y^i_{m_1,m_2} > {}^*Q^i_{m,0}.$$

But $^*Q^i_{m,0} \geq {}^*y^i_{m,0}$ by definition. Therefore,

$$\sum_{(m_1,m_2)\in A'_i | m_1-m_2=m, m_1\neq m} {}^*y^i_{m_1,m_2} > 0.$$

That is, there exists at least one $m' > m$ such that $^*y^i_{m',m'-m} > 0$.

Now, since the inflow into node m is at least $\alpha_{im}\epsilon$, it follows by flow balance that the outflow from node m, $\sum_{(n|n\in K'_i, 0\leq n<m)} {}^*Q^i_{m,n} + {}^*s_{im} \geq \alpha_{im}\epsilon$. Therefore, $\sum_{(n|n\in K'_i, 0<n<m)} {}^*Q^i_{m,n} + {}^*s_{im} \geq \alpha_{im}\epsilon - {}^*Q^i_{m,0} \geq \beta_{im}\epsilon - {}^*Q^i_{m,0} > 0$, where the last two inequalities follow from Definition 6 and assumption ($^*Q^i_{m,0} < \beta_{im}\epsilon$), respectively. Consequently, $\sum_{(n|n\in K'_i, 0<n<m)} {}^*Q^i_{m,n}$ and $^*s_{im}$ can not both be zero and we have the following cases:

Case 1): Suppose $^*s_{im} > 0$. Consider the following alternative scheme:

(a) Reduce $^*y^i_{m',m'-m} > 0$ by some small amount ν, thus saving $\nu\delta_i$ units in cutting costs. This reduces production of size m by ν units.

(b) Increase ${}^*y^i_{m,0}$ by ν with <u>no</u> added cutting cost. This increases production of size m by ν units.

(c) Compensate for the extra outflow from node m by decreasing scrap ${}^*s_{im} > 0$ by ν units.

(d) Compensate for the decreased outflow from node m' by increasing scrap ${}^*s_{im'}$ by ν.

<u>Case 2)</u>: Suppose ${}^*s_{im} = 0$, so that $\sum_{(n|n \in K'_i, 0<n<m)} {}^*Q^i_{m,n} > 0$. Without loss of generality, suppose for some node $0 < n < m$, ${}^*Q^i_{m,n} > 0$. Consider the following alternative scheme:

(1) Reduce ${}^*y^i_{m',m'-m}$ by some small amount ν, thus saving $\nu\delta_i$ units in cutting costs. This reduces production of size m by ν units.

(2) Increase ${}^*y^i_{m,0}$ by ν with <u>no</u> added cutting cost. This increases production of size m by ν units.

(3) Compensate for the extra outflow from node m by decreasing ${}^*Q^i_{m,n}$ by ν units and save at least $\nu\delta_i$ units of cutting costs.

(4) Compensate for the decreased outflow from node m' by increasing ${}^*Q^i_{m',m'-(m-n)}$ by ν units, and add cutting costs exactly equal to those saved in Step (3).

Comparing either new cutting scheme with the original one, we find: 1) Total production of sizes are the same in each scheme; 2) Either of the new cutting schemes will save costs from the original cutting scheme. This contradicts the assumption that the original cutting scheme is optimal. Thus, ${}^*Q^i_{m,0} \geq \beta_{im}\epsilon$. □

We are now ready to prove Proposition 1.

Proof of Proposition 1: Given $m \in \Delta_0$, ${}^*y^i_{m,0} \leq \beta_{im}\epsilon$ since m is a scrap size. Suppose ${}^*y^i_{m,0} < \beta_{im}\epsilon$. Since ${}^*Q^i_{m,0} \geq \beta_{im}\epsilon$ by Lemma 1, it follows that there is at least one path $p \in P^i_{m,0}$ with intermediate nodes between node m and node 0 along which there are positive flows. Consider one such path with a flow of ${}^*q^i_p$ units. Assume the path consists of b arcs that produce ${}^*q^i_p$ pieces of sizes $k_1, k_2, ..., k_b$ where $\sum_{j=1}^{b} k_j = m$. Since all of these sizes are scrap sizes and the total demands for these are $\beta_{i(m-k_1)}\epsilon$, ..., and $\beta_{ik_b}\epsilon$, it follows that ${}^*q^i_p \leq \min_{1\leq j\leq b}(\beta_{ik_j}\epsilon)$. Consider ν such that $0 < \nu \leq {}^*q^i_p$, and the following alternative scheme:

(a) Reduce the flow along the path by ν units and increase ${}^*s_{im}$ by ν units, thus save $(b-1)\nu\delta_i$ units in cutting costs.

(b) Increase ${}^*y^i_{m-k_j,0}$ for $j = 1, 2, ..., b$ by ν units and add <u>no</u> cutting cost (note that this is always possible since $\alpha_{i(m-k_j)}\epsilon > \beta_{i(m-k_j)}\epsilon \geq \nu$ for each node in the path).

This will save $(b-1)\nu\delta_i$ units of cutting costs and satisfy all demands, which contradicts the assumption of optimality. Therefore, ${}^*y^i_{m,0} = \beta_{im}\epsilon$ and the additional demand of $\beta_{im}\epsilon$ units for size m are satisfied at no additional cost. The remainder of $(\alpha_{im}\epsilon - \beta_{im}\epsilon)$ units is scrapped at node m. $\square$

Now suppose optimal flows in the original and the perturbed problem are given by the vectors $\hat{y}$ and ${}^*y$ respectively, while the corresponding optimal values of the objectives are given by $\hat{z}$ and ${}^*z$. Assuming that cutting costs are strictly positive and that we are not charged for the additional inflows of $\alpha_{im}\epsilon$, it is clear that the optimal solution to the original problem with the following modifications will be an optimal solution to the perturbed problem:

- For $m \in \Delta_1$, the extra inflow of $\alpha_{im}\epsilon$ units is routed along arc $(m, 0)$, i.e., ${}^*y_{m,0} = \hat{y}_{m,0} + \alpha_{im}\epsilon = \hat{y}_{m,0} + \beta_{im}\epsilon$. Thus it satisfies the additional demand of $\beta_{im}\epsilon$ units for size m at no additional cost since there is no cutting.
- For $m \in \Delta_0$, a portion $\beta_{im}\epsilon$ of the extra inflow of $\alpha_{im}\epsilon$ units is routed along arc $(m, 0)$, i.e., ${}^*y_{m,0} = \hat{y}_{m,0} + \beta_{im}\epsilon = \beta_{im}\epsilon$. Thus it satisfies the additional demand of $\beta_{im}\epsilon$ units for size m at no additional cost. The remainder of $(\alpha_{im}\epsilon - \beta_{im}\epsilon)$ units is scrapped at node m since there is no original demand for size m.

Thus we do not pay any additional cutting costs in the perturbed problem and ${}^*z = \hat{z} - \sum_{i\in I, m\in\Delta_0}(\alpha_{im}\epsilon - \beta_{im}\epsilon)\sigma_i$. We can see that, as $\varepsilon \to 0$ this ε-perturbation reduces to the original problem.

To clarify and illustrate the perturbation and its optimal solution, consider facility i with $S_i = \{11\}$ and $D = \{3, 5\}$ shown in Figure 2. Suppose that $\lambda_3 = 20$ and $\lambda_5 = 10$ and the optimum solution has $\hat{y}_{11,6} = \hat{y}_{6,3} = \hat{y}_{3,0} = 10$ as shown in Figure 4.

For the ϵ-perturbation we have $\Delta = \{1, 2, 3, 5, 6, 8, 11\}$ with $\Delta_0 = \{1, 2\}$ and $\Delta_1 = \{3, 5, 6, 8, 11\}$, and we define $\alpha_{im} = \beta_{im}$ for $m \in \Delta_1$ and $\alpha_{im} > \beta_{im}$ for $m \in \Delta_0$. The

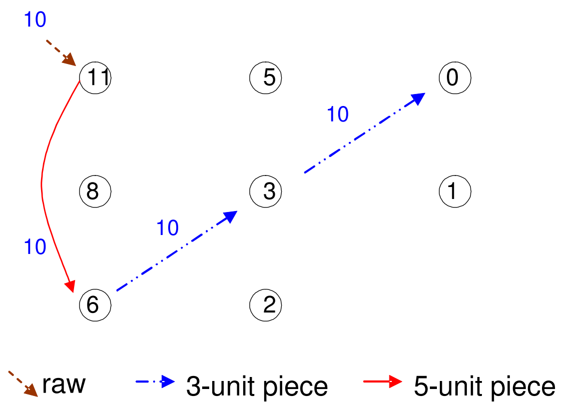

Figure 4: Optimal Cutting Scheme for Original Problem

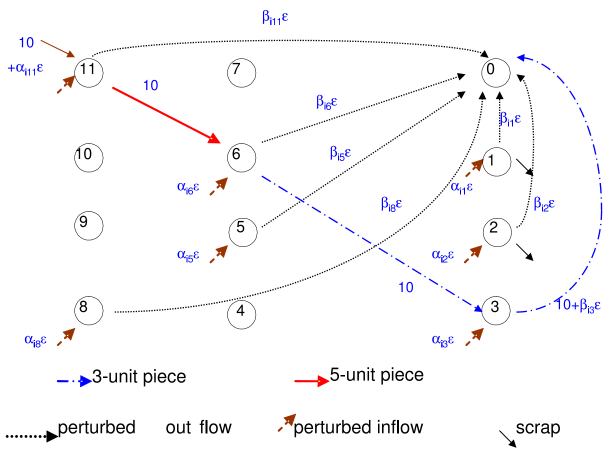

Figure 5: Optimal Cutting Scheme for Perturbed Problem

node set for the perturbed problem is given by $K_i' = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11\}$, and the optimal flow for the perturbed problem described above is given by Figure 5.

While the above flows describe the optimum for the perturbed problem, they are not necessarily unique. To illustrate this fact, consider the flow pattern depicted in Figure 6, which differs from Figure 5 in the following flows only:

- ${}^*y_{8,0}$, which is now equal to zero,

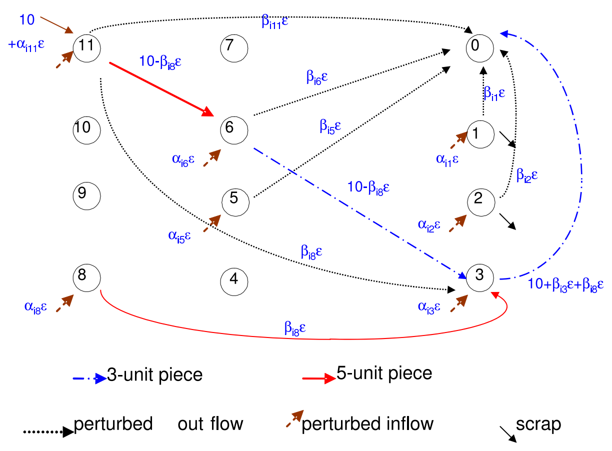

Figure 6: Alternative Optimum for Perturbed Problem

- ${}^*y_{11,6}$, ${}^*y_{6,3}$, each of which is decreased by an amount $\beta_{i8}\epsilon$, and
- ${}^*y_{11,3}$, ${}^*y_{8,3}$, ${}^*y_{3,0}$, each of which is increased by an amount $\beta_{i8}\epsilon$.

Note that flow balance is maintained and the net flow for each different size is identical in both cases so that all demand is satisfied. We save $2(\beta_{i8}\epsilon)\delta_i$ corresponding to reduced flows in arcs ${}^*y_{11,6}$ and ${}^*y_{6,3}$ that now need not be cut, but pay an additional $2(\beta_{i8}\epsilon)\delta_i$ due to the increased flows along ${}^*y_{11,3}$ and ${}^*y_{8,3}$ and the corresponding extra cutting cost. Thus, there is no overall change in costs and thus this also represents an optimal solution.

Rather than solving the perturbed problem directly, we will find it is convenient to construct an optimal solution from the optimal solution to the unperturbed problem.

We also present an algorithm for constructing alternative optima when these exist. Algorithm 1 constructs an optimal solution to the perturbed problem:

Algorithm 1 Obtaining an Optimal Solution to the Perturbed Primal Problem

```
1: Solve the unperturbed problem and obtain the optimal solution $\hat{y}$, with optimal value $\hat{z}$.
2: For $m \in \Delta_1$, let ${}^*y_{m,n} = \hat{y}_{m,n}$ for $n > 0$ and ${}^*y_{m,0} = \hat{y}_{m,0} + \alpha_{im}\epsilon = \hat{y}_{m,0} + \beta_{im}\epsilon$.
3: For $m \in \Delta_0$, let ${}^*y_{m,0} = \beta_{im}\epsilon$ and scrap the remainder of $(\alpha_{im}\epsilon - \beta_{im}\epsilon)$ units at node $m$.
```

Proposition 2. *The solution obtained by Algorithm 1 is optimal for the perturbed problem.*

Proof. For $m \in \Delta_0$, the conclusion follows from Step 2 of Algorithm 1 and Proposition 1.

For $m \in \Delta_1$, we create exactly as much of supply ($\alpha_{im}\epsilon$) as the new demand ($\beta_{im}\epsilon$). By assigning all of the extra supply to the arc $(m, 0)$, we create exactly enough of size m to satisfy all additional demands at no extra cost, from which the conclusion follows. □

Corollary 1. *Let $\hat{z}$ be the optimal value for the unperturbed problem. Then the optimal value ${}^*z$ for the perturbed problem is: ${}^*z = \hat{z} - \sum_{i\in I, m\in\Delta_0}(\alpha_{im}\epsilon - \beta_{im}\epsilon)\sigma_i$.*

Suppose that in the optimal solution created by Algorithm 1, $\exists m, m' \in \Delta_1, k_1, k_2, ..., k_p \in D$ with $k_1 + k_2 + ... + k_p = m, m' > m$ such that ${}^*y^i_{m',m'-k_1} > 0$, ${}^*y^i_{m'-k_1,m'-k_1-k_2} > 0$, ..., ${}^*y^i_{m'-\sum_{a=1}^{a=p-1}k_a,m'-m} > 0$. Then we can find alternative optimal solutions for the perturbed problem by Algorithm 2:

Algorithm 2 Obtaining Alternative Optimal Solutions to the Perturbed Primal Problem

```
1: Apply Algorithm 1 and obtain an optimal solution $^*y$.
2: Reduce the flow along the path $(m', m' - k_1), (m' - k_1, m' - k_1 - k_2), ..., (m' - \sum_{a=1}^{a=p-1} k_a, m' - m)$ by some small amount $\nu \leq \beta_{im}\epsilon$.
3: Compensate for the reduced production of sizes $k_1, k_2, ..., k_p$ by increasing the flow along the path $(m, m - k_1), (m - k_1, m - k_1 - k_2), ..., (m - \sum_{a=1}^{a=p-1} k_a, 0)$ by $\nu$.
4: Compensate for the increased flow out of node $m$ by reducing the flow along arc $(m, 0)$ by $\nu$, thus producing $\nu$ fewer pieces of size $m$.
5: Compensate for the decreased flow out of node $m'$ by increasing the flow along the arc $(m', m' - m)$ by $\nu$, thus producing $\nu$ additional pieces of size $m$.
```

We now show that the solution obtained by Algorithm 2 is also optimal.

Proposition 3. *Suppose that in the optimal solution created by Algorithm 1, $\exists m, m' \in \Delta_1 (m' > m), k_1, k_2, ..., k_p \in D$ with $k_1 + k_2 + ... + k_p = m$ such that ${}^*y^i_{m',m'-k_1} > 0$, ${}^*y^i_{m'-k_1,m'-k_1-k_2} > 0$, ..., ${}^*y^i_{m'-\sum_{a=1}^{a=p-1}k_a,m'-m} > 0$. Then by applying Algorithm 2 for $\forall m \in \Delta_1$, we can create alternative optimal solutions to the perturbed problems at each facility $i, i \in I$ with the following properties*

$\forall m \in \Delta_1$,

- *${}^*y^i_{m',m'-m} > 0$;*
- *${}^*y^i_{m,m-k_1} > 0$, ${}^*y^i_{m-k_1,m-k_1-k_2} > 0$, ..., ${}^*y^i_{m-\sum_{a=1}^{a=p-1}k_a,0} > 0$.*

Proof. In Step 2 of Algorithm 2, we save νp units in cutting costs.

In Step 3 of Algorithm 2, we pay $\nu(p-1)$ units in cutting costs.

In Step 4 of Algorithm 2, no cutting costs are added.

In Step 5 of Algorithm 2, we pay ν units in cutting costs.

Thus Algorithm 2 incurs exactly the same cutting costs as Algorithm 1 while producing the same amount of products. Consequently, the solution created by Algorithm 2 is optimal. $\square$

Note that Figure 5 illustrates an optimal solution by Algorithm 1 while Figure 6 illustrates an alternative optimal solution by Algorithm 2 with $p = 2$, $m = 8, m' = 11$ and $k_1 = 3, k_2 = 5$. The following proposition characterizes any solution to the perturbed problem.

Proposition 4. *$\forall i \in I, m \in \Delta_1$, any solution to the perturbed problem must have the following characteristics,*

a) ${}^*y^i_{m,0} \geq \beta_{im}\epsilon > 0$, or

b) $0 \leq {}^*y^i_{m,0} < \beta_{im}\epsilon$, $\exists m' \in K_i, m' > m$, and $k_1, k_2, ..., k_q \in K_i$ with $\sum_{g=1}^{q} k_g = m$ such that

- ${}^*y^i_{m',m'-m} > 0$;
- ${}^*y^i_{m,m-k_1} > 0$, ${}^*y^i_{m-k_1,m-k_1-k_2} > 0$, ..., ${}^*y^i_{(m-\sum_{g=1}^{q-1} k_g),0} > 0$.

Proof. $\forall m \in \Delta_1$: From Lemma 1, ${}^*Q^i_{m,0} = \sum_{p \in P^i_{m,0}} {}^*q^i_p = {}^*y^i_{m,0} + \sum_{p \in (P^i_{m,0} \setminus (m,0))} {}^*q^i_p \geq \beta_{im}\epsilon$. We consider the two cases: ${}^*y^i_{m,0} \geq \beta_{im}\epsilon$ and ${}^*y^i_{m,0} < \beta_{im}\epsilon$.

a) If ${}^*y^i_{m,0} \geq \beta_{im}\epsilon > 0$, the result follows immediately.

b) Suppose ${}^*y^i_{m,0} < \beta_{im}\epsilon$ so that $\sum_{p \in (P^i_{m,0} \setminus (m,0))} {}^*q^i_p > 0$. Then there must exist at least one path with flow amount ν such that $0 < \nu \leq \beta_{im}\epsilon - {}^*y^i_{m,0}$ and $k_1, k_2, ..., k_p$ with $k_1 + k_2 + ... + k_p = m$ such that ${}^*y^i_{m,m-k_1} > 0$, ${}^*y^i_{m-k_1,m-k_1-k_2} > 0$, ..., ${}^*y^i_{(m-\sum_{g=1}^{p-1} k_g),0} > 0$.

Furthermore, since ${}^*y^i_{m,0} < \beta_{im}\epsilon$, it follows that $\exists m' > m$ such that ${}^*y^i_{m',m'-m} > 0$ to satisfy the additional demand for size m. $\square$

To summarize the need for the preceding description, we note that because of dual degeneracy we may obtain inherent values that may not satisfy the properties of monotonicity and superadditivity, which makes a price-directed policy inapplicable. We therefore develop and solve a perturbed model. In the next section, we show that the inherent values from the perturbed model will satisfy these describe properties; Section 2.4 will be used for this purpose.

2.4 PROPERTIES OF THE ϵ-PERTURBATION

We now prove some important properties of the static problem that will hold under an ϵ-perturbation, and thus make our policy applicable in a dynamic environment.

Proposition 5. *The perturbed problem is always feasible.*

Proof. The conclusion follows from the fact that if right hand sides of constraints (2.2) and (2.4) are positive, the constraints can always be satisfied by adjusting the raw material consumption because there are no upper bounds for these: there are no capacity limits at facilities. □

Denote an optimal primal solution to the perturbed problem by $({}^*y, {}^*r, {}^*s)$ and the corresponding optimal dual multipliers by $({}^*\pi, {}^*\mu, {}^*\eta)$. Recall that we also assume that the cutting cost δ_i is strictly positive.

The following proposition proves that permutability holds, i.e. satisfying a demand for size k_1 by cutting node m and then cutting the remnant to satisfy a demand for size k_2, is equivalent to satisfying the demand for size k_2 first and then k_1. Interested readers may refer to the paper by Adelman and Nemhauser [2] for a discussion of permutability property when cutting costs are zero.

Proposition 6. *For any $i \in I, k_1, k_2 \in \Delta$, and $m, m-k_1, m-k_1-k_2 \in K_i'$, suppose ${}^*y^i_{m,m-k_1} > 0$ and ${}^*y^i_{m-k_1,m-k_1-k_2} > 0$. Then*

a) ${}^*\pi_{ik_1} = {}^*\eta_{i,m} - {}^*\eta_{i,m-k_1} + \delta_i$, ${}^*\pi_{ik_2} = {}^*\eta_{i,m-k_1} - {}^*\eta_{i,m-k_1-k_2} + \delta_i$.

b) ${}^*\pi_{ik_2} = {}^*\eta_{i,m} - {}^*\eta_{i,m-k_2} + \delta_i$, ${}^*\pi_{ik_1} = {}^*\eta_{i,m-k_2} - {}^*\eta_{i,m-k_2-k_1} + \delta_i$.

Proof. The conclusion of part a) directly follows from complementary slackness corresponding to the assumption that there exist ${}^*y^i_{m,m-k_1} > 0$ and ${}^*y^i_{m-k_1,m-k_1-k_2} > 0$.

To show the conclusion of part b), by complementary slackness corresponding to (2.8), it follows that

$$ {}^*\pi_{ik_1} = {}^*\eta_{i,m} - {}^*\eta_{i,m-k_1} + \delta_i, \tag{2.20} $$

and

$$ {}^*\pi_{ik_2} = {}^*\eta_{i,m-k_1} - {}^*\eta_{i,m-k_1-k_2} + \delta_i. \tag{2.21} $$

Constraint (2.8) for arcs $(m, m-k_2)$ and $(m-k_2, m-k_1-k_2)$ implies that

$$ {}^*\eta_{i,m} - {}^*\eta_{i,m-k_2} + \delta_i \geq {}^*\pi_{ik_2}, \tag{2.22} $$

and

$$ {}^*\eta_{i,m-k_2} - {}^*\eta_{i,m-k_1-k_2} + \delta_i \geq {}^*\pi_{ik_1}. \tag{2.23} $$

(2.21) and (2.22) imply that

$$ {}^*\eta_{i,m-k_2} \leq {}^*\eta_{i,m} - {}^*\eta_{i,m-k_1} + {}^*\eta_{i,m-k_1-k_2}, \tag{2.24} $$

while (2.20) and (2.23) imply that

$$ {}^*\eta_{i,m-k_2} \geq {}^*\eta_{i,m} - {}^*\eta_{i,m-k_1} + {}^*\eta_{i,m-k_1-k_2}. \tag{2.25} $$

Then, from (2.24) and (2.25) it follows that

$$ {}^*\eta_{i,m-k_2} = {}^*\eta_{i,m} - {}^*\eta_{i,m-k_1} + {}^*\eta_{i,m-k_1-k_2}. \tag{2.26} $$

Equation (2.26) can then be written in the following two ways using (2.21) and (2.20) respectively:

$$ {}^*\eta_{i,m} - {}^*\eta_{i,m-k_2} + \delta_i = {}^*\eta_{i,m-k_1} - {}^*\eta_{i,m-k_1-k_2} + \delta_i = {}^*\pi_{ik_2}, $$

and

$$ {}^*\eta_{i,m-k_2} - {}^*\eta_{i,m-k_1-k_2} + \delta_i = {}^*\eta_{i,m} - {}^*\eta_{i,m-k_1} + \delta_i = {}^*\pi_{ik_1}, $$

from which the conclusion follows. □

The following proposition assigns an appropriate inherent value to scrap sizes.

Proposition 7. *Under an ϵ-perturbation, $\forall i \in I, m \in \Delta_0, {}^*\eta_{i,m} = m\sigma_i$.*

Proof. The total inflow into node m is $\sum_{(n,m)\in A'_i} {}^*y^i_{n,m} + \alpha_{im}\epsilon$ and it is equal to the outflow from node m, $\sum_{(m,n)\in A'_i} {}^*y^i_{m,n} + {}^*s_{im}$. Consider ${}^*y^i_{m,n}, \forall (m,n) \in A'_i$. From Proposition 1, ${}^*y^i_{m,0} = \beta_{im}\epsilon$. Further, ${}^*y^i_{m,n} = 0$ for all other n since $m-n < m$ has no original demand. Thus, $\sum_{(m,n)\in A'_i} {}^*y^i_{m,n} + {}^*s_{im} = \beta_{im}\epsilon + {}^*s_{im} = \sum_{(n,m)\in A'_i} {}^*y^i_{n,m} + \alpha_{im}\epsilon \geq \alpha_{im}\epsilon$. But by Definition 6, $\beta_{im}\epsilon < \alpha_{im}\epsilon$. It follows that ${}^*s_{im} > 0$. The result then follows from complementary slackness corresponding to (2.11). $\square$

Proposition 9, which we introduce next, shows that δ-superadditivity will always hold with the ϵ-perturbation. To prove this proposition, we first introduce the following crucial proposition, which will also be used subsequently to prove δ-monotonicity.

Proposition 8. *Under an ϵ-perturbation, $\forall i \in I, m \in K_i, {}^*\pi_{im} = {}^*\eta_{i,m}$.*

Proof. Consider an optimal primal solution ${}^*y$ to the perturbed problem.

For $m \in \Delta_0$, ${}^*y^i_{m,0} > 0$ by Proposition 1. Complementary slackness corresponding to (2.9) implies that ${}^*\pi_{im} = {}^*\eta_{i,m} - {}^*\eta_{i,0}$. By Proposition 7, ${}^*\eta_{i,0} = 0$, and thus, ${}^*\pi_{im} = {}^*\eta_{i,m}$.

For node $m \in \Delta_1$, there are two possibilities:

Case 1): ${}^*y^i_{m,0} > 0$. The proof is identical to the proof for $m \in \Delta_0$.

Case 2): ${}^*y^i_{m,0} = 0$. From Proposition 4, $\exists m' > m$ such that ${}^*y^i_{m',m'-m} > 0$ to satisfy the additional demand for size m. Let l' denote a raw size, from which there is a flow into node m'.

Proposition 4 also states that there exists a positive flow along the path $p = \{(m, m-k_1), (m-k_1, m-k_1-k_2), ..., (m - \sum_{g=1}^{q-1} k_g, 0)\}$, from which we obtain products with sizes $k_1, k_2, ..., k_q$. Note that sizes $k_1, k_2, ..., k_q$ must correspond to sizes that have original demand.

From Corollary 1, it follows that the optimal solution to the perturbed problem must not use additional original raw material, that is, the optimal solution to the perturbed problem uses the same amount of all original raw material as the one to the unperturbed problem. Thus, flow along the path p must free up exactly the same amount (say ν units) of the same size of raw material (i.e. raw size l') as the extra requirement of the flow along the arc $(m', m' - m)$. To ensure this, there must exist m'' such that ${}^*y^i_{m'',m''-k_1} > 0$, ${}^*y^i_{m''-k_1,m''-k_1-k_2} > 0$, ..., ${}^*y^i_{(m''-\sum_{g=1}^{q-1} k_g),m''-m} > 0$ in the path from l' to m. Note that m'' may or may not be identical to node m'.

By complementary slackness for (2.8), the following q equalities follow

$$
\begin{gathered}
{}^*\eta_{i,m''} - {}^*\eta_{i,m''-k_1} + \delta_i = {}^*\pi_{ik_1}. \\
{}^*\eta_{i,m''-k_1} - {}^*\eta_{i,m''-k_1-k_2} + \delta_i = {}^*\pi_{ik_2}. \\
\vdots
\end{gathered}
$$

$$
{}^*\eta_{i,(m''-\sum_{g=1}^{q-1} k_g)} - {}^*\eta_{i,m''-m} + \delta_i = {}^*\pi_{ik_q}. \tag{2.27}
$$

Adding all equations of the form (2.27), we obtain

$$
{}^*\eta_{i,m''} - {}^*\eta_{i,m''-m} + q\delta_i = {}^*\pi_{ik_1} + \ldots + {}^*\pi_{ik_q}. \tag{2.28}
$$

Also, ${}^*y^i_{m,m-k_1} > 0$, ..., ${}^*y^i_{(m-\sum_{g=1}^{q-2} k_g),(m-\sum_{g=1}^{q-1} k_g)} > 0$ and complementary slackness corresponding to (2.8) imply that

$$
\begin{gathered}
{}^*\eta_{i,m} - {}^*\eta_{i,m-k_1} + \delta_i = {}^*\pi_{ik_1}, \\
\vdots
\end{gathered}
$$

$$
{}^*\eta_{i,(m-\sum_{g=1}^{q-2} k_g)} - {}^*\eta_{i,(m-\sum_{g=1}^{q-1} k_g)} + \delta_i = {}^*\pi_{ik_{q-1}}, \tag{2.29}
$$

while ${}^*y^i_{(m-\sum_{g=1}^{q-1} k_g),0} > 0$ and complementary slackness corresponding to (2.9) imply that

$$
{}^*\eta_{i,(m-\sum_{g=1}^{q-1} k_g)} - {}^*\eta_{i,0} = {}^*\pi_{ik_q}. \tag{2.30}
$$

Adding all equations of the form (2.29) to equation (2.30) and recalling that ${}^*\eta_{i,0} = 0$ from Proposition 7, it follows that

$$
{}^*\eta_{i,m} + (q-1)\delta_i = {}^*\pi_{ik_1} + \ldots + {}^*\pi_{ik_q}. \tag{2.31}
$$

Finally, ${}^*y^i_{m',m'-m} > 0$ and complementary slackness corresponding to (2.8) imply that

$$
{}^*\eta_{i,m'} - {}^*\eta_{i,m'-m} + \delta_i = {}^*\pi_{im}. \tag{2.32}
$$

By Proposition 6, we obtain

$$^*\eta_{i,m''} - ^*\eta_{i,m''-m} + \delta_i =^* \eta_{i,m'} - ^*\eta_{i,m'-m} + \delta_i = ^*\pi_{im}. \tag{2.33}$$

Comparing equations (2.28) and (2.33), we conclude

$$^*\pi_{im} + (q-1)\delta_i = ^*\pi_{ik_1} + ... + ^*\pi_{ik_q}. \tag{2.34}$$

From equations (2.31) and (2.34), it then follows that

$$^*\pi_{im} = ^*\eta_{i,m}, \tag{2.35}$$

thus concluding the proof. □

This leads to the following proposition:

Proposition 9. *Let $(^*\eta, ^*\pi)$ be optimal dual multipliers under an ϵ-perturbation. Then the value function $^*\eta$ is δ-superadditive.*

Proof. Consider any nodes $l, m, l{+}m \in K_i$ at an arbitrary facility $i \in I$. From dual constraint (2.8), $^*\eta_{i,l+m} + \delta_i - ^*\eta_{i,m} \geq ^*\pi_{il}$, and by Proposition 8, $^*\pi_{il} = ^*\eta_{i,l}$. Thus, $^*\eta_{i,l+m} + \delta_i \geq ^*\eta_{i,l} + ^*\eta_{i,m}$. □

Next, we address δ-monotonicity and show that it follows as a corollary from the following proposition, which provides a general relationship between the inherent values of different sizes at a facility.

Proposition 10. *Let $(^*\eta, ^*\pi)$ be optimal dual multipliers under an ϵ-perturbation. Then $^*\eta_{i,m} + \delta_i - \sigma_i(m-n) \geq ^*\eta_{i,n}$, $\forall i \in I, m, n \in K_i$ such that $m - n \in K_i$.*

Proof. $^*\eta_{i,m} + \delta_i - (m-n)\sigma_i \geq ^*\eta_{i,m} + \delta_i - ^*\eta_{i,m-n} \geq ^*\pi_{i,n}$ where the first inequality follows from dual constraint (2.11), and the second follows from constraint (2.8). By Proposition 8, $^*\pi_{i,n} = ^*\eta_{i,n}$, from which the proposition follows. □

Proposition 10 indicates that the inherent value of a remnant depends on both cutting costs and scrap value. Two corollaries follow from this: Corollary 2 shows that as long as cutting costs are positive, δ-monotonicity follows directly, while Corollary 3 shows that if the unit scrap value at each facility is higher than the unit cutting cost at the facility, then the inherent value is strictly decreasing with size.

Corollary 2. *Let $(^*\eta, ^*\pi)$ be optimal dual multipliers under an ϵ-perturbation. Then the value function $^*\eta$ is strictly δ-monotonic, that is, $^*\eta_{i,m} + \delta_i > {}^*\eta_{i,n}$, $\forall i \in I, m, n(m > n) \in K_i$.*

Proof. Follows directly from Proposition 10 and the fact that $(m-n)\sigma_i > 0$. $\square$

Corollary 3. *Under an ϵ-perturbation, $\forall i \in I$, if $\sigma_i \geq \delta_i$, then $^*\eta_{i,m} \geq {}^*\eta_{i,n}$, $\forall m > n, m, n \in K_i$.*

Proof. If $\sigma_i \geq \delta_i$, then since $m - n \geq 1$, $\delta_i - \sigma_i(m-n) \leq 0$. Then, Proposition 10 implies that $^*\eta_{i,m} \geq {}^*\eta_{i,n}$. $\square$

Conversely, if the cutting cost is high, it is possible that $(m-n)\sigma_i - \delta_i \leq \eta_{i,m} - \eta_{i,n} < 0$, so that the inherent values could actually increase as the size decreases.

The next proposition that we introduce provides an upper bound on the inherent value of a specific size at a specific facility. Suppose that for any given raw stock of size h and remnant of size l, we denote $d_{h,l}$ as the maximum number of pieces of the remnant that can be cut from one piece of raw stock, $d'_{h,l}$ as the number of cuts made to obtain $d_{h,l}$ pieces, and $b_{h,l}$ as the scrap size remaining after cutting. That is, $d_{h,l} = \lfloor h/l \rfloor, b_{h,l} = h - ld_{h,l}$, and $d'_{h,l} = d_{h,l}$ if $b_{h,l} > 0$, $d'_{h,l} = d_{h,l} - 1$ otherwise. Then we obtain a bound on the inherent value as follows:

Proposition 11. *Under an ϵ-perturbation, $\forall i \in I, l \in K_i$, $^*\eta_{i,l} \leq \min_{(h \in S_i, h \geq l)}\{a_{ih} + d'_{h,l}\delta_i - b_{h,l}\sigma_i\}/d_{h,l}$.*

Proof. Consider an arc from node h $(h \geq l)$ to node $m = h - l$, where $h \geq l$. By constraint (2.8),

$$^*\eta_{i,h} + \delta_i \geq {}^*\eta_{i,h-l} + {}^*\pi_{il}.$$

Adding $(d'_{h,l} - 1)\delta_i$ to both sides yields

$$^*\eta_{i,h} + d'_{h,l}\delta_i \geq {}^*\eta_{i,h-l} + {}^*\pi_{il} + (d'_{h,l} - 1)\delta_i.$$

Since ${}^*\eta_{i,h} \le a_{ih}$ from (2.10), and ${}^*\pi_{il} = {}^*\eta_{i,l}$ from Proposition 8, it follow that

$$a_{ih} + d'_{h,l}\delta_i \ge {}^*\eta_{i,h-l} + {}^*\eta_{i,l} + (d'_{h,l} - 1)\delta_i.$$

Now consider the arc from node $h - l$ to node $h - 2l$. Adding ${}^*\eta_{i,l} + (d'_{h,l} - 2)\delta_i$ to both sides of constraint (2.8) for this arc and using Proposition 8 we obtain

$${}^*\eta_{i,h-l} + {}^*\eta_{i,l} + (d'_{h,l} - 1)\delta_i \ge {}^*\eta_{i,h-2l} + 2{}^*\eta_{i,l} + (d'_{h,l} - 2)\delta_i.$$

If we continue the same procedure through nodes $h - 2l$, $h - 3l$, ... we eventually reach the arc from node $l + b_{h,l}$ to node $b_{h,l}$, where size $b_{h,l}$ is either zero or corresponds to a scrap size. The constraint for this arc (constraint (2.8) if $b_{h,l} > 0$ or constraint (2.9) if $b_{h,l} = 0$) yields

$${}^*\eta_{i,h-(d_{h,l}-1)l} + (d_{h,l} - 1){}^*\eta_{i,l} + (d'_{h,l} - (d_{h,l} - 1))\delta_i \ge {}^*\eta_{i,b_{h,l}} + d_{h,l}{}^*\eta_{i,l}.$$

The above sequence of relationships implies that

$$a_{ih} + d'_{h,l}\delta_i \ge {}^*\eta_{i,b_{h,l}} + d_{h,l}{}^*\eta_{i,l}.$$

By Proposition 7, ${}^*\eta_{i,b_{h,l}} = b_{h,l}\sigma_i$, it follows that

$$a_{ih} + d'_{h,l}\delta_i \ge b_{h,l}\sigma_i + d_{h,l}{}^*\eta_{i,l},$$

from which the proposition follows. □

Based upon computational tests, it appears that on average, these bounds are between approximately 7.10% (for small problems) and 11.32% (for large problems) higher than the true values. Details may be found in Section 2.5.

Finally, the following proposition provides a bound on the difference in inherent values between the same sizes but at different facilities:

Proposition 12. *If the scrap value is the same for each facility, then under ϵ-perturbation,* $|{}^*\eta_{i,l} - {}^*\eta_{i',l}| \le \min_{h\in S_i, h'\in S_{i'}}\{a_{ih} - h\sigma + \delta_i, a_{i'h'} - h'\sigma + \delta_{i'}\}$.

Proof. Consider any raw size h, and any remnant l ($(h,l) \in A_i$) at facility i ($i \in I$). From dual constraint (2.8), it follows that

$$^*\eta_{i,l} \leq a_{ih} - {}^*\eta_{i,h-l} + \delta_i. \tag{2.36}$$

constraint (2.11) and the above inequality yield

$$^*\eta_{i,l} \leq a_{ih} - (h-l)\sigma + \delta_i. \tag{2.37}$$

If we consider a similar constraint for size l at facility i', it follows that

$$^*\eta_{i',l} \leq a_{i'h'} - (h'-l)\sigma + \delta_{i'}. \tag{2.38}$$

From dual feasibility, we obtain

$$l\sigma \leq {}^*\eta_{i',l}, \tag{2.39}$$

and

$$l\sigma \leq {}^*\eta_{i,l}. \tag{2.40}$$

From (2.37) and (2.39), we obtain

$$^*\eta_{i,l} - {}^*\eta_{i',l} \leq a_{ih} - h\sigma + \delta_i. \tag{2.41}$$

Similarly, (2.38) and (2.40) imply that

$$^*\eta_{i',l} - {}^*\eta_{i,l} \leq a_{i'h'} - h'\sigma + \delta_{i'}. \tag{2.42}$$

From (2.41) and (2.42), the result then follows. □

To check the performance of our policy based on the inherent values, we develop a simulation model to compare our policy with two other policies. The results are described in the following chapter.

2.5 SIMULATION AND INVENTORY CONTROL

We designed and ran a simulation model to compare our policy with two other heuristic policies to test how our adaptation of the optimal static policy performs in a dynamic and stochastic demand environment. We assume this supply chain is operated over a long period of time. In each period (day), demand for various sizes arise at various demand location. These demands arrive randomly according to independent Poisson processes with rate λ_{jk} for size k at location j. There are no capacity constraints on the facility and all demand is to be fulfilled.

Our goal was to test our price-directed policy against two other plausible heuristic policies. Each of the three policies is described briefly below:

A. Smallest Fit: This is a greedy approach where we search all facilities for the smallest remnant or raw piece that can accommodate the demanded size. Ties are broken by picking the option that minimizes the sum of raw materials costs (if any), shipping costs, and cutting costs, less the value of any scrap generated.

B. Multi-criteria Heuristic: This policy follows a hierarchy of choices. First, if a remnant of the exact size demanded exists, it is used. If not, we look for the smallest feasible remnant that can not be used after cutting the current demand size. If no such piece exists, we cut from the largest feasible remnant in stock (thus leaving another smaller useful remnant behind). Finally, if there are no feasible remnants, we cut from the largest raw stock size. In all instances, ties are broken using the same criterion as in Policy A.

C. Price-Directed Policy: This is our policy that uses dual values from the perturbed version of the static LP formulation. We look across all facilities and pick the facility i and size m according to the criterion described in Theorem 1, where ties are broken arbitrarily. An important consideration with a price-directed policy is that inventories must be suitably controlled; without constraints on raw materials it is possible that inventories of certain remnant sizes can grow unchecked. Adelman and Nemhauser [2] get around this by empirically setting a suitable (and identical) limit on the inventory of each size.

We follow a different approach. For each day, we track the number of pieces of all sizes created, as well as the number of pieces of each raw size used during that day at every site.

These numbers are then compared to the optimal $y^i_{m,n}$ and r_{ih} values from the solution to the static LP in order to set limits on their inventory as follows: suppose the demand to be met is for size k, so that cutting a piece from the selected size m would leave a remnant of size $n = m - k$. If the size m corresponds to a remnant we check to see if the total number of pieces of size n created after this operation exceeds the optimal daily rate from the static LP for size n (i.e., exceeds the sum of the flow into or out of node n). If the size m corresponds to a raw size we check to see if the usage of that raw size exceeds the optimal daily rate r_{im}. If either of these occurs we forego the option of using facility i and size m, and move on to the next best one according to the inherent value criterion. The only time we violate these inventory restrictions is if it is completely impossible to otherwise satisfy the demand. In that case we just pick the location and raw size that yield the best value for the inherent value criterion.

To compare these policies we tested three randomly generated sets of instances using our simulation model: these consist of 10 small, 10 medium and 10 large-size instances respectively. Table 4 provides some key characteristics of each instance set. For each of the thirty instances, the actual demand for each size at each demand location over a 300-day period was generated randomly according to a Poisson process, and an overall time-ordered vector of demands was first created, with each element having a demand location, a size and a time associated with it. The same vector was then separately used as the input to each of the three different policies. For each instance studied, the simulation was run for 300 days with statistics collected after day 50. Ten replications were run for each instance, with each replication using a different randomly generated demand vector, thus resulting in a total of 100 instances for each instance set, each of which was analyzed in turn using the three heuristics. Total costs for a given instance were averaged across the ten replications. The cost elements were the same as those in the LP formulation, namely raw materials, cutting, and shipping. Revenues from all recovered scrap were subtracted from the total costs.

For a complete list of computational results, see Appendix A. Table 5 summarizes the results. For every instance, the price-directed policy was the least expensive. For clarity of exposition, the costs for the other two policies were re-scaled by defining the cost for the price-directed policy for an instance to be 1.0, i.e., the numbers in the table represent

Table 4: Test Instance Characteristics

Size of Instances in set	*No. of Facilities*	*No. of Customers*	*Unique Raw Sizes*	*Unique Demand Sizes*	*No. of Nodes in Network*
Small	2	2 to 4	3 to 4	4 to 6	14 to 22
Medium	4 to 5	10 to 20	5 to 13	7 to 38	50 to 384
Large	10 to 20	40 to 75	15 to 29	19 to 99	352 to 3097

Table 5: Performance of the Policies*

Size of Instances in Set	*Smallest Fit*			*Multi-criteria Heuristic*		
	Min.	*Max.*	*Average*	*Min.*	*Max.*	*Average*
Small	1.18	1.55	1.39	1.04	1.30	1.16
Medium	1.13	1.78	1.47	1.16	1.74	1.37
Large	1.51	1.98	1.72	1.37	2.17	1.66

*Costs for the two policies in the above table represent multipliers of the corresponding cost for the price-directed policy.

multipliers of the corresponding cost for the price-directed policy. The entries in Table 5 represent the average of these costs across the ten different instances in each instance set, along with the maximum and minimum costs. The results indicate that on average, the multi-criteria heuristic can result in costs that are from 16 to 66% higher, while a naive policy such as the greedy heuristic (smallest fit) can cost from 39 to 72% more. In the worst case, the costs could be 98 to 117% more. These results clearly indicate the benefits of a price-directed policy that adapts the optimal results of the static LP formulation to a stochastic and dynamic environment.

Finally, while conducting these computational experiments, we also gathered data on the inherent values ($\eta_{i,k}$) so as to evaluate the quality of the bounds that were developed in Proposition 11. The value of each $\eta_{i,k}$ was compared to the theoretical upper bound from Proposition 11 and the gap expressed as a percentage of the value was then averaged across all $\eta_{i,k}$ values for each instance. Detailed computational results can be found Chapter A in Appendix. Table 6 below summarizes our observations:

Table 6: Performance of Bounds Given by Proposition 11

Size of Instances in Set	*Percentage Gap*		
	Min.	*Max.*	*Average*
Small	2.19	12.04	7.10
Medium	4.99	13.00	8.49
Large	8.38	14.93	11.32

As might be expected, the bounds are tighter for the smaller instances. However, even for the larger instances, they are not unreasonably large. This suggests that as a quick and dirty approach, one could use the bound from Proposition 11 for the inherent values and still expect to perform quite reasonably.

3.0 LOCATION MODEL

The LP model in Chapter 2 can be extended to include a facility location problem. We are given a set of candidate distribution centers with different fixed costs and capacities at the different locations, and we may choose not to operate facilities at one or more of these locations. In this section, we integrate the production, location and distribution problems together. Compared with the LP model discussed in Chapter 2, the situation considered in this section is further complicated by the fact that in the design phase we must choose which distribution centers to open, and in the operational phase we must choose a particular open distribution center and a particular size to be cut at that facility. The operational decision is also complicated by the fact that we may need several different centers to meet all demand because of the capacity limits at these facilities, and each of these could have several different feasible sizes for cutting; the latter is true because open distribution centers also maintain inventories of potentially useful remnants generated from prior cutting operations. Thus, the design decision influences both the ability to satisfy demand at a later point in time, as well as the associated long-run cost. To solve this problem, the basic model in Chapter 2 is expanded to include location as a decision variable. The objective in this chapter is to determine which facilities should be operated, along with a long-term policy for cutting and distributing the product in order to satisfy demand.

3.1 PROBLEM FORMULATION AND POLICY DEVELOPMENT

We consider the same production-distribution system described in Chapter 2, but now assume that facility i has a fixed cost f_i and processing capacity M_i. The problem is to determine which facilities to open, along with a long-term policy for how to cut and distribute the product to satisfy dynamic and ongoing demand.

3.1.1 Primal Formulation of the Static Problem

In addition to the notation introduced in Section 2.2 of Chapter 2, we make the following additional definitions:

- f_i = Equivalent uniform daily fixed cost of operating a facility at location $i \in I$. We uniform the total fixed cost based on unit operation time to make it comparative with the other operating costs we defined in the previous chapter. This ensures that the fixed cost will not dominate other daily operating costs and the model has a proper objective function.
- M_i = Total daily processing capacity for facility $i \in I$.
- z_i = Binary variable to decide whether facility i should be operated or not; $z_i = 1$, if facility $i \in I$ is opened; $z_i = 0$, otherwise.

We are now ready to formulate the problem as the following mixed-integer program:
Program P-MIP

$$\min \sum_{i\in I}\sum_{h\in S_i} a_{ih}r_{ih} + \sum_{i\in I}\sum_{\{(m,n)\in A_i|n>0\}} \delta_i y^i_{m,n} + \sum_{i\in I}\sum_{j\in J}\sum_{k\in D_j} c_{ijk}x_{ijk} + \sum_{i\in I} f_i z_i - \sum_{i\in I}\sum_{u\in K_i} u\sigma_i s_{iu} \tag{3.1}$$

subject to:

$$\left(r_{il} + \sum_{m|(m,l)\in A_i} y^i_{m,l}\right) - \left(s_{il} + \sum_{n|(l,n)\in A_i} y^i_{l,n}\right) = 0, \text{for all } i \in I \text{ and } l \in K_i, \tag{3.2}$$

$$\sum_{i\in I} x_{ijk} = \lambda_{jk}, \text{ for all } j \in J \text{ and } \ k \in D_j, \tag{3.3}$$

$$\left(\sum_{(m,n)\in A_i|(m-n)=k} y^i_{m,n}\right) - \left(\sum_{j\in J} x_{ijk}\right) = 0, \text{ for all } i \in I \text{ and } k \in D, \tag{3.4}$$

$$\sum_{j\in J}\sum_{k\in D_j} x_{ijk} \le M_i z_i, \text{ for all } i \in I, \tag{3.5}$$

$$x, y, r, s \ge 0; z_i \in \{0,1\}. \tag{3.6}$$

Note that this formulation is identical to the one in Chapter 2 except for the additional term $\sum_{i\in I} f_i z_i$ in the objective function and constraint (3.5), which ensures that no shipments take place from a facility that is not chosen for operation and that total shipments out of an operational facility i cannot exceed its processing capacity. We assume that the above problem always has a feasible solution.

Proposition 13. *The problem described by Program P-MIP is NP-hard.*

Proof. Observe that if we simplify our problem by assuming that D and S are identical (i.e., all demand is only for sizes that are available as raws) we can satisfy all demand without any cutting. Then Program P-MIP contains the multi-product capacitated facility location problem (MPCFL) as a special case (for a discussion of MPCFL, the reader is referred to [63, 64]). Since MPCFL is known to be NP-hard [75], it follows that P-MIP is NP-hard. □

3.1.2 Implementation in a Dynamic Environment

In Section 3.1.1, we have presented a formulation for a *static* version of our problem, i.e., we have assumed a fixed value for the demand rate for each size at each location. In practice, demand is stochastic. We assume that it follows a Poisson process with rate vector λ. To manage the long-run cost under a dynamic environment, we need to make decisions over two stages.

In the first stage, the decision on which facility to open is made. Once this decision has been made, the capacity and fixed costs are known, and the static problem reduces to the linear program of Chapter 2, and at this point, we could follow the procedure described therein for the second stage.

Let us first consider the above P-MIP but for a fixed binary location vector $z = \hat{z}$. This yields the following linear programming problem:

Program P-SP($\hat{z}$)

$$\min \sum_{i\in I}\sum_{h\in S_i} a_{ih} r_{ih} + \sum_{i\in I}\sum_{\{(m,n)\in A_i|n>0\}} \delta_i y^i_{m,n} + \sum_{i\in I}\sum_{j\in J}\sum_{k\in D_j} c_{ijk} x_{ijk} - \sum_{i\in I}\sum_{u\in K_i} u\sigma_i s_{iu} + \sum_{i\in I} f_i \hat{z}_i \quad (3.7)$$

Subject to (3.2), (3.3), (3.4), and

$$\sum_{j \in J} \sum_{k \in D_j} x_{ijk} \leq M_i \hat{z}_i, \quad \text{for all} \;\; i \in I, \tag{3.8}$$

$$x, y, r, s \geq 0. \tag{3.9}$$

To obtain the dual corresponding to P-SP($\hat{z}$) we define dual variables as follows:

- $\eta_{i,m}$ corresponding to each constraint in (3.2), i.e., to each size at each facility,
- μ_{jk} corresponding to each constraint in (3.3), i.e., to each demanded size at each location,
- π_{ik} corresponding to each constraint in (3.4), i.e., to each demanded size produced at each facility,
- ω_i corresponding to each constraint in (3.8), i.e., to each facility.

Here, ω_i may be interpreted as the marginal cost of one extra unit of processing capacity at facility i. By dropping the constant term $\sum_{i \in I} f_i \hat{z}_i$ from the objective of P-SP($\hat{z}$), we can write out its dual formulation as follows:

Program D-SP($\hat{z}$)

$$\text{Maximize} \quad \sum_{j \in J} \sum_{k \in D_j} \lambda_{jk} \mu_{jk} - \sum_{i \in I} M_i \hat{z}_i \omega_i \tag{3.10}$$

subject to:

$$\mu_{jk} - \pi_{ik} - \omega_i \leq c_{ijk}, \text{ for all } i \in I, j \in J, k \in D_j, \tag{3.11}$$

$$\eta_{i,n} - \eta_{i,m} + \pi_{ik} \leq \delta_i, \text{ for all } i \in I; m, n \in K_i; (m,n) \in A_i; k = m - n, n > 0, \tag{3.12}$$

$$\eta_{i,n} - \eta_{i,m} + \pi_{ik} \leq 0, \text{ for all } i \in I; m, n \in K_i; (m,n) \in A_i; k = m - n, n = 0, \tag{3.13}$$

$$\eta_{i,h} \leq a_{ih}, \text{ for all } i \in I, h \in S_i, \tag{3.14}$$

$$\eta_{i,u} \geq u\sigma_i, \text{ for all } i \in I, u \in K_i, \tag{3.15}$$

$$\omega_i \geq 0, \text{ for all } i \in I, \tag{3.16}$$

$$\eta, \mu, \pi \text{ unrestricted.}$$

Consider Program P-SP($\hat{z}$) and let $\hat{I}$ be the set of open facilities such that $\hat{z}_i = 1$ for all $i \in \hat{I} \subseteq I$ and $\hat{z}_i = 0$ for $i \notin \hat{I}$. If the capacity constraint given by (3.8) did not exist, it was shown in Chapter 2 that in the optimal solution to the resulting (static) LP problem, all demand for a particular size k at a particular location j (namely, λ_{jk}) will be satisfied from a single demand location $^*i$ by cutting from a piece of size $^*m$, where $(^*i, ^*m) \in argmin_{i \in I, (m | m \in K_i, m \geq k)}(^*\eta_{i,m} - {}^*\eta_{i,m-k} + \delta_i + c_{ijk})$.

For the capacitated version of the LP that we consider herein, the situation is a little more complicated since it might happen that demand for size k at location j is satisfied from more than one facility since the "best" option might correspond to a facility with insufficient capacity. Let $(^*x, ^*y, ^*r, ^*s)$ be an optimal solution to P-SP($\hat{z}$), and $(^*\eta, ^*\mu, ^*\pi, ^*\omega)$ be the corresponding optimal dual vector. Suppose λ^i_{jk} is the amount of demand for size k at location j that is satisfied by facility i in the optimal solution to P-SP($\hat{z}$). Note that $\sum_{i \in \hat{I}} \lambda^i_{jk} = \lambda_{jk}$. Let $^*\lambda^i_{jk}$ be the corresponding amount at the optimum. If all demand λ_{jk} happens to come from a single facility at the optimum of Program P-SP($\hat{z}$), a result similar to that discussed above holds. Proposition 14 below states and proves this.

Proposition 14. *If $\sum_{i \in \hat{I}} \lambda^i_{jk} = \lambda^{i'}_{jk} = \lambda_{jk}$ for some $i' \in \hat{I}$, the optimal static solution will fulfill all of the demand λ_{jk} by choosing the facility i' and cutting the size $m'(\geq k)$, where $(i', m') \in argmin_{i \in \hat{I}, (m | m \in K_i, m \geq k)}(^*\eta_{i,m} - {}^*\eta_{i,m-k} + \delta_i + c_{ijk})$.*

Proof. Consider the complementary slackness condition corresponding to (3.8):

$$(M_i \hat{z}_i - \sum_{j \in J} \sum_{k \in D_j} {}^*x_{ijk})^*\omega_i = 0, \quad \text{for all} \quad i \in I. \tag{3.17}$$

When a facility $i \in \hat{I}$ is open and its capacity is not fully used, that is, $\sum_{j \in J} \sum_{k \in D_j} {}^*x_{ijk} < M_i \hat{z}_i$, then (3.17) implies $^*\omega_i = 0$. Let $I_1 (I_1 \subset \hat{I})$ be the set of open facilities whose capacities are fully used and $I_2 = \hat{I} \backslash I_1$. Thus, $\sum_{i \in I_2} M_i (^*\omega_i \hat{z}_i) = 0$.

For $i \in I_1$, $M_i = \sum_{j \in J} \sum_{k \in D_j} {}^*x_{ijk} = \sum_{j \in J} \sum_{k \in D_j} {}^*\lambda^i_{jk}$. Now consider the optimal value of the objective function of Program D-SP($\hat{z}$):

$$\begin{aligned}
&\textstyle\sum_{j \in J} \sum_{k \in D_j} {}^*\mu_{jk} \lambda_{jk} - \sum_{i \in I} M_i (^*\omega_i \hat{z}_i), \\
&= \textstyle\sum_{j \in J} \sum_{k \in D_j} {}^*\mu_{jk} \lambda_{jk} - \sum_{i \in I_1} {}^*\omega_i M_i, \\
&= \textstyle\sum_{i \in \hat{I}} \sum_{j \in J} \sum_{k \in D_j} {}^*\lambda^i_{jk} {}^*\mu_{jk} - \sum_{i \in I_1} \sum_{j \in J} \sum_{k \in D_j} {}^*\lambda^i_{jk} {}^*\omega_i, \\
&= \textstyle\sum_{i \in \hat{I}} \sum_{j \in J} \sum_{k \in D_j} {}^*\lambda^i_{jk} {}^*\mu_{jk} - \sum_{i \in I_1} \sum_{j \in J} \sum_{k \in D_j} {}^*\lambda^i_{jk} {}^*\omega_i - \sum_{i \in I_2} \sum_{j \in J} \sum_{k \in D_j} {}^*\lambda^i_{jk} {}^*\omega_i, \\
&= \textstyle\sum_{i \in \hat{I}} \sum_{j \in J} \sum_{k \in D_j} {}^*\lambda^i_{jk} {}^*\mu_{jk} - \sum_{i \in \hat{I}} \sum_{j \in J} \sum_{k \in D_j} {}^*\lambda^i_{jk} {}^*\omega_i, \\
&= \textstyle\sum_{i \in \hat{I}} \sum_{j \in J} \sum_{k \in D_j} {}^*\lambda^i_{jk} (^*\mu_{jk} - {}^*\omega_i).
\end{aligned}$$

Now consider the set $I_k = \{i \in \hat{I} | \exists m \geq k, {}^*y^i_{m,m-k} > 0\}$ at optimality. That is, I_k is the set of all open facilities that produce units of demand size k at the optimum. We would like to see which of these facilities could possibly have ${}^*x_{ijk} > 0$ at optimality, i.e., fulfill the demand arising at location j for size k. Suppose then that $i' \in I_k$ is such a facility, i.e., ${}^*x_{i'jk} > 0$ at optimality. From constraint (3.11) of the dual problem we know that ${}^*\mu_{jk} - {}^*\omega_i \leq {}^*\pi_{ik} + c_{ijk}$ for all $i \in \hat{I}$. Since (a) ${}^*\mu_{jk} - {}^*\omega_i$ appears only in (3.11), (b) ${}^*\lambda^i_{jk} \geq 0$, and (c) the dual objective is maximized, it therefore follows that

$$ {}^*\mu_{jk} - {}^*\omega_i = \min_{i \in \hat{I}} ({}^*\pi_{ik} + c_{ijk}). \tag{3.18} $$

Now consider complementary slackness conditions corresponding to (3.11) and (3.12):

$$ ({}^*\mu_{jk} - {}^*\omega_i - {}^*\pi_{ik} - c_{ijk}){}^*x_{ijk} = 0, \quad \text{for all} \;\; i \in \hat{I}, j \in J, k \in D_j. \tag{3.19} $$

$$ ({}^*\eta_{i,n} - {}^*\eta_{i,m} - \delta_i + {}^*\pi_{ik}){}^*y^i_{m,n} = 0, \;\text{for all}\; i \in \hat{I}, (m,n) \in A_i, k = m - n. \tag{3.20} $$

Since ${}^*x_{i'jk} > 0$ by assumption, complementary slackness condition (3.19) implies that ${}^*\mu_{jk} - {}^*\omega_{i'} = {}^*\pi_{i'k} + c_{i'jk}$, so that from (3.18), $i' \in argmin_{i \in \hat{I}}({}^*\pi_{ik} + c_{ijk})$. Also, from complementary slackness condition (3.20), for all $i \in I_k$, if size m' is cut down to size $n = (m' - k)$, (so that ${}^*y^i_{m',n} > 0$), it follows that ${}^*\pi_{ik} = {}^*\eta_{i,m'} - {}^*\eta_{i,m'-k} + \delta_i$. Since all defined $c_{ijk} > 0$ it follows from the two previous sentences that $(i', m') \in argmin_{i \in \hat{I}, (m \in K_i | m \geq k)}({}^*\eta_{i,m} - {}^*\eta_{i,m-k} + \delta_i + c_{ijk})$. □

For the case where λ_{jk} is fulfilled from more than one location, it is clear that there is at least one facility ${}^*i$ and size ${}^*m$ for which Proposition 14 holds; we can use the same argument as we did in proving Proposition 14. There may be other facilities though, where Proposition 14 will not necessarily hold. However, as with the uncapacitated case this leads us to an intuitively appealing policy that we can adopt for the dynamic environment. When a demand arrives, we pick the facility and size according the criterion of Proposition 14. If the facility's capacity has been exhausted we remove it from consideration and pick the next best option based on the same criterion, and so on. It is worth noting that in the actual dynamic and stochastic demand situation, even if the capacity of the best facility has not been exhausted we might still have to move on to the next best option because the selected size ${}^*m$ might not be in stock at the facility ${}^*i$ at the time when that demand arrives.

A second factor that needs to be kept in mind when trying to adapt the static optimum to the dynamic problem is the fact that once again, not all of the optimal dual vectors can be used with a price-directed policy such as the one described above, since they do not all have appealing properties such as monotonicity or superadditivity described in Chapter 2. For the uncapacitated problem this difficulty was addressed by defining and solving a suitable perturbation, where each of the right-hand sides for some of the constraints are increased by a small amount. This ensured a unique set of optimal multipliers that also have properties that are desirable from the perspective of a price-directed policy. Since none of the proofs for this in Chapter 2 are affected by the capacity constraints given by (3.8), we follow an identical approach here as well. The perturbation of Program P-SP($\hat{z}$), that we solve is exactly the same as the one solved in the uncapacitated case, with the exception of the additional capacity constraints.

In summary, the procedure followed for our location-distribution problem is to a) solve Program P-MIP and obtain the optimal values ${}^*x_{ijk}$ and ${}^*z_i$, b) solve the perturbed version of Program P-SP(${}^*z$) in order to obtain the optimal dual vector, and c) use these dual values and the criterion of Proposition 14 to satisfy demand as it arrives.

To solve P-MIP we use an approach based on Benders' decomposition. There are two reasons for doing this. First, it is a well known procedure that has been shown to work well on such problems (e.g., [40, 108, 112]). Second, it can often obviate the need to define and solve a perturbation when the solution obtained is non-degenerate. In such cases one can directly use the values of the dual multipliers that are obtained from the solution to Benders' subproblem at the last (optimal) iteration; with general IP solvers, one does not have direct access to dual information and after solving the P-MIP, we would have to solve a perturbation of Program P-SP(${}^*z$) to obtain the multipliers.

3.2 BENDERS' DECOMPOSITION

Let Ω and Ψ denote the sets of extreme points and extreme rays respectively of the dual polyhedron described by (3.11-3.16). Benders (1962) proposed a solution methodology, whereby the decision variables for the above problem are partitioned into "hard" and "easy" variables (the integer z variables and the continuous ones, respectively). Since the optimal value of Program P-SP($\hat{z}$) is equal to the optimal value of Program D-SP($\hat{z}$) described in Section 3.1.2 (plus the constant $\sum_{i \in I} f_i \hat{z}_i$), we may rewrite the original problem (P-MIP) as the following master problem:

Program MP

$$\min \sum_{i \in I} f_i z_i + \theta \tag{3.21}$$

subject to:

$$\theta \geq \sum_{j \in J} \sum_{(k \in D | \lambda_{jk} > 0)} \lambda_{jk} \mu_{jk} - \sum_{i \in I} M_i \omega_i z_i, (\mu, \eta, \pi, \omega) \in \Omega, \tag{3.22}$$

$$\sum_{i \in I} M_i \omega_i z_i \geq \sum_{j \in J} \sum_{(k \in D | \lambda_{jk} > 0)} \lambda_{jk} \mu_{jk}, (\mu, \eta, \pi, \omega) \in \Psi, \tag{3.23}$$

$$z_i \in \{0,1\}. \tag{3.24}$$

Note that θ represents the optimal value of Program D-SP($\hat{z}$) and (3.23) is required to ensure that enough capacity is built to satisfy demand. If we consider only a subset of the extreme points in Ω (say $\Omega' \subseteq \Omega$), and a subset of the extreme rays in Ψ (say say $\Psi' \subseteq \Psi$), then the above program may be approximated via the following restricted master problem:

Program RMP(Ω', Ψ')

$$\min \sum_{i \in I} f_i z_i + \theta \tag{3.25}$$

subject to:

$$\theta \geq \sum_{j \in J} \sum_{k \in D} \lambda_{jk} \mu_{jk} - \sum_{i \in I} M_i \omega_i z_i, \quad (\mu, \eta, \pi, \omega) \in \Omega', \tag{3.26}$$

$$\sum_{i \in I} M_i \omega_i z_i \geq \sum_{j \in J} \sum_{k \in D} \lambda_{jk} \mu_{jk}, \quad (\mu, \eta, \pi, \omega) \in \Omega' \Psi', \tag{3.27}$$

$$z_i \in \{0,1\}. \tag{3.28}$$

The constraints (3.26) and (3.27) are referred to as Benders' optimality and feasibility cuts, respectively. Program RMP is a 0-1 integer program in z with the exception of the variable θ. Essentially, Benders' decomposition algorithm alternates between solving a restricted master problem (Program RMP) and a linear subproblem (Program D-SP(z)) at each iteration. Program RMP is solved for z with an extra constraint added on at each successive iteration. Program D-SP(z) is solved using a fixed z (obtained from the current approximation RMP) and is in turn used to generate a constraint that is then added to Program RMP at the next iteration. The basic algorithm may be described as follows:

Step 1: Set $L = 0$ and $U = \infty$ as lower and upper bounds on the true optimum; set $r = s = t = p = 0$ and pick any binary vector $z = {}^*z$ and go to Step D.

Step 2: Solve Program RMP to obtain the optimal vector $z = {}^*z$. Update $L = \theta + \sum_{i \in I} {}^*z_i f_i$.

Step 3: If $L = U$ stop.

Step 4: Solve D-SP(${}^*z$) to obtain either the optimal vector $\{{}^*\mu, {}^*\eta, {}^*\pi, {}^*\omega\}$ if it has an optimum solution, or if the problem is unbounded, obtain the extreme direction $\{{}^*\mu, {}^*\eta, {}^*\pi, {}^*\omega\}$ along which it is unbounded. If it is the former case, go to Step 5, if it is the latter case go to Step 6.

Step 5: Set $r = r + 1, t = t + 1, \mu^t = {}^*\mu$ and $\omega^t = {}^*\omega$; generate an optimality cut to add the constraint $\Theta \geq \sum_{j \in J} \sum_{k \in D} \lambda_{jk} \mu^t_{jk} - \sum_{i \in I} M_i \omega^t_i z_i$, to RMP. Update $U = \sum_{i \in I} f_i {}^* z_i + \sum_{j \in J} \sum_{k \in D} \lambda_{jk} \mu^t_{jk} - \sum_{i \in I} M_i \omega^{t*}_i z_i$ and go to Step 2.

Step 6: Set $s = s + 1, p = p + 1, \mu^p = {}^*\mu$ and $\omega^p = {}^*\omega$; generate a feasibility cut to add the constraint $\sum_{i \in I} M_i \omega^p_i z_i \geq \sum_{j \in J} \sum_{k \in D} \lambda_{jk} \mu^p_{jk}$ to RMP. Go to Step 2.

3.3 GUARANTEEING FEASIBILITY OF P-SP($\hat{Z}$)

At each iteration of Benders' procedure, we either add a cut corresponding to (3.26) that results in a tighter bound for the master problem, or add a cut corresponding to (3.27) that restricts the feasible region of the master problem in order to bound the subproblem D-SP($\hat{z}$) generated by $\hat{z}$, or equivalently, ensuring that P-SP($\hat{z}$) is feasible. With large problems there could be many such cuts, and this could considerably increase the time required to solve the problem. Furthermore, since the procedure generates the cut from information at the current iteration, it will often generate several cuts dominated by other cut generated later before solving the problem. In light of these points, we examine the possibility of improving the algorithm by adding feasibility constraints to the master problem directly in order to guarantee that the objective of subproblem D-SP$(\hat{z})$ is bounded, i.e., that P-SP$(\hat{z})$ is feasible.

We now develop two versions of these constraints that guarantee the feasibility of the primal subproblem and illustrate them with examples. We first present the feasibility constraints based on unique demand sizes in the following section.

3.3.1 Demand Size Version

Without loss of generality, suppose there are e elements in the demand set D, $D = (k_1, k_2, ..., k_e)$, such that:

$$k_1 < k_2 < k_3 < ... < k_e.$$

Define $I_{k_g} \subset I, g = 1, 2, ..., e$ as the set of the facilities such that, each facility in this set has at least one raw size equal to or longer than k_g. Thus, facilities in the set I_{k_g} have feasible raw materials for the demand size k_g. Note that $I_{k_1} = I$.

Theorem 2. *Subproblem P-SP(z) will be feasible if, and only if, vector z satisfies the following constraints:*

$$\sum_{i \in I_{k_g}} M_i z_i \geq \sum_{j \in J} \sum_{l=g}^{e} \lambda_{jk_l}, \forall g = 1, 2, ..., e. \tag{3.29}$$

Proof. For necessity, note that if P-SP(z) is feasible for the vector z, then all demand is satisfied and thus, facilities must have enough capacity to satisfy each demanded size. Therefore, constraints (3.29) must be satisfied by the vector z.

For sufficiency, consider a vector z that satisfies all constraints (3.29). For a particular demand size k it is clear that demand can be satisfied if the following two conditions are met:

i) One or more open facilities have raw materials that are long enough to satisfy this demand size.

ii) The open facilities have sufficient processing capacity to satisfy the demand for this size.

Now suppose the constraints specified by (3.29) hold.

For a given h, the left-hand side of each constraint represents the total open capacities at all facilities that process raw sizes that are longer than or equal to r_h. On the other hand, the right-hand side is the total demand for all sizes k that are longer than or equal to r_h. For all such k it is obvious that i) holds and (3.29) guarantees that ii) will be true as well. Thus, the demand for each size k can be satisfied. Since this holds for all h, it is clear that all demand can be satisfied and hence P-SP(z) is feasible. □

To illustrate how these constraints work and explore interesting characteristics among them, we give an example. In this example, there are two facility locations and three demand locations. At each location, there are some raw and demand sizes. The detailed information is listed in Table 7.

Table 7: Parameters of Raw and Demand Sizes

	Location 1			Location 2			Location 3			
Raw Size	7	-	-	7	10	12	-	-	-	-
Demand Size	2	4	8	2	8	11	2	4	8	11
Amount	10	20	10	20	30	10	20	10	30	40

From the above table, we can see that there four unique demand sizes: $k_1 = 2, k_2 = 4, k_3 = 8, k_4 = 11$. Thus, $e = 4$. Correspondingly, we obtain four sets of facilities such that facilities in each set have raw size feasible for each demand size. We list these four sets as follows

$$I_{k_1} = I_2 = (1, 2),$$

$$I_{k_2} = I_4 = (1, 2),$$

$$I_{k_3} = I_8 = (2),$$

$$I_{k_4} = I_{11} = (2),$$

For each demand size, we have to develop a constraint corresponding to constraint (3.29) to ensure all demand for this size will be satisfied. For total four demand sizes 2, 4, 8, 11, we need the following four constraints:

$$\sum_{i \in I_{k_1}} M_i z_i = M_1 z_1 + M_2 z_2 \geq \sum_{j \in J} \sum_{l=1}^{4} \lambda_{jk_l} = 200. \tag{3.30}$$

$$\sum_{i \in I_{k_2}} M_i z_i = M_1 z_1 + M_2 z_2 \geq \sum_{j \in J} \sum_{l=2}^{4} \lambda_{jk_l} = 150. \tag{3.31}$$

$$\sum_{i \in I_{k_3}} M_i z_i = M_2 z_2 \geq \sum_{j \in J} \sum_{l=3}^{4} \lambda_{jk_l} = 120. \tag{3.32}$$

$$\sum_{i \in I_{k_4}} M_i z_i = M_2 z_2 \geq \sum_{j \in J} \sum_{l=4}^{4} \lambda_{jk_l} = 50. \tag{3.33}$$

We can rewrite the above four constraints as follows

$$M_1 z_1 + M_2 z_2 \geq 200. \tag{3.34}$$

$$M_1 z_1 + M_2 z_2 \geq 150. \tag{3.35}$$

$$M_2 z_2 \geq 120. \tag{3.36}$$

$$M_2 z_2 \geq 50. \tag{3.37}$$

It is not hard to see that although constraints (3.29) can guarantee feasibility, some constraints may be dominated by others. In the example we just presented, among all 4 constraints, two of them are dominated by other two. Generally, if there exist two demand sizes $k_1, k_2 (k_1 < k_2)$ such that $r_g < k_1, k_2 \leq r_{g+1}$, then constraint (3.29) corresponding to demand size k_2 is always a dominated cut. The reason behind this is that if constraint (3.29) corresponding to demand size k_1 is satisfied, then constraint (3.29) corresponding to demand size k_2 must be satisfied. To avoid adding dominated feasibility constraints into the master problem and improve the computational efficiency, we address more compact feasibility constraints based on raw sizes in the following section.

3.3.2 Raw Size Version

Suppose there are t elements in the raw set S, $S = (r_1, r_2, ..., r_t)$, where r_t represents the size of the t^{th} raw. Define $S' = (r_0, r_1, r_2, ..., r_t)$ where $r_0 = 0$. Without loss of generality, we assume the elements in S' are ordered:

$$r_0 < r_1 < r_2 < ... < r_t.$$

Define $I_{r_h} \subset I, h = 0, ..., t-1$ as the set of all facilities that have at least one raw size longer than r_h. Observe that $I_{r_0} = I$ and that all demands with size k such that $r_h < k \leq r_{h+1}$ must be satisfied by raw sizes $r_{h+1}, ..., r_t$. The following proposition ensures the feasibility of subproblem P-SP(z).

Proposition 15. *The primal subproblem (Program P-SP(z)) with the vector z obtained from the master problem (Program MP) is feasible if, and only if, z satisfies all of the following constraints:*

$$\sum_{i \in I_{r_h}} M_i z_i \geq \sum_{j \in J} \sum_{(k \in D_j, k > r_h)} \lambda_{jk}, \forall h \in (0, 1, ..., t-1). \tag{3.38}$$

Before we prove this proposition, we first present an example to illustrate it. In this example, we consider an example with two facilities and two demand locations. The raw sizes available at each facility along with the unit costs are summarized in Table 8. This table also provides the unit salvage values and unit cutting cost at each facility. The demanded sizes and the corresponding demand rates at each demand point, along with unit shipping costs are summarized in Table 9.

Table 8: Facility Characteristics

Facility 1				Facility 2			
Raw Size	Cost	Unit Scrap Value	Cutting Cost	Raw Size	Cost	Unit Scrap Value	Cutting Cost
13 7	20 12	0.05	0.05	7	13	0.1	0.06

In this example, there are 2 original raw sizes. After adding 0 to the raw set, we obtain $S' = (0, 7, 13)$. Namely, $r_0 = 0, r_1 = 7, r_2 = 13$. The corresponding facility set feasible for $r_h, h = 0, 1$ is as follows

$$I_{r_1} = I_7 = (1),$$

$$I_{r_0} = I_0 = (1, 2).$$

Demand for size $k = 9$ can only be met by raw size 13 from facility in the set I_7, while other demand sizes, i.e. 3, 5, 6, 7 can be met by raw sizes 7 and 13 from facilities in the set I_0.

Then, since there are 2 original unique raw sizes, we have two feasibility constraints. We generate them as follows

$$\sum_{i \in I_{r_1}} M_i z_i = M_1 z_1 \geq \sum_{j \in J} \sum_{(k \in D_j, k > r_1)} \lambda_{jk} = 25, \tag{3.39}$$

$$\sum_{i \in I_{r_0}} M_i z_i = M_1 z_1 + M_2 z_2 \geq \sum_{j \in J} \sum_{(k \in D_j, k > r_0)} \lambda_{jk} = 120. \tag{3.40}$$

Or, we can rewrite them as

$$M_1 z_1 \geq 25, \tag{3.41}$$

Table 9: Parameters of Demands

	Location 1		Location 2			
Size	5	6	3	5	7	9
Demand	20	20	10	30	15	25
Facility 1	2	4	2	5	6	7
Facility 2	3	3	1.5	3	4	5

$$M_1 z_1 + M_2 z_2 \geq 120. \tag{3.42}$$

Now, let us check whether these two constraints can guarantee the feasibility of the problem. First, let us consider constraint (3.41). Obviously, if it is satisfied, facility 1 must be opened and the demand for size 9 can be satisfied by raw size 13.

Second, consider constraint (3.42). This constraint ensures that Facility 1 and 2 together can produce are at least 120 units. In other words, in addition to the capacity required to satisfy the demand for size 9, the open facilities can still produce at least 95 units. Since the total demand for all sizes 3, 5, 6, 7 is 95, no matter whether we open only Facility 1 or both facilities, the open facilities always have raw materials long enough to satisfy demand for sizes 3, 5, 6, 7. Thus, all demand for sizes 3, 5, 6, 7 can always be satisfied.

Now, we are ready to prove Proposition 15 as follows:

Proof of Proposition 15: To show necessity, note that if P-SP(z) is feasible for the vector z obtained from the master problem, then all demand is satisfied and therefore there must be enough capacity to process the raw sizes required to satisfy each demanded size. Therefore constraints (3.38) must be satisfied by the vector z.

For sufficiency, consider a vector z obtained from the master problem that satisfies all constraints specified by (3.38). Then P-SP(z) will be feasible if all demand can be met for this vector. Now suppose the constraints specified by (3.38) hold.

For a given h, the left-hand side of each constraint represents the total open capacities at all facilities that process raw sizes that are longer than or equal to r_{h+1}. On the other hand, the right-hand side is the total demand for all sizes k that are longer than r_h. For all such k it is obvious that (3.38) guarantees that open facilities have enough feasible raw sizes to satisfy the demand. Thus, the demand for each size $k \in (r_{t-1}, r_t]$ can be satisfied. Since this holds for all h it is clear that all demand can be satisfied and hence P-SP(z) is feasible, and thus, replacing constraints (3.23) by constraints (3.38) in the master problem (Program MP) guarantees that the primal subproblem with z will be feasible. $\square$

Next, we show that replacing constraints (3.23) in the master problem with the constraints specified by (3.38) maintains its equivalence to the original problem, i.e., it does not eliminate any feasible points in the original problem (Program P-MIP). To show this, we first prove the following lemma.

Lemma 2. *If z is feasible for Program P-MIP, then it satisfies constraints (3.38).*

Proof. From constraint (3.3), it follows that

$$\sum_{j\in J}\sum_{(k\in D_j,k>r_h)}\lambda_{jk} = \sum_{j\in J}\sum_{(k\in D_j,k>r_h)}\sum_{i\in I}x_{ijk} = \sum_{i\in I}\sum_{j\in J}\sum_{(k\in D_j,k>r_h)}x_{ijk}, \forall h = 0,1,...,t-1. \tag{3.43}$$

Recalling the definition of I_{r_h}, it follows that

$$\sum_{i\in I}\sum_{j\in J}\sum_{(k\in D_j,k>r_h)}x_{ijk} = \sum_{i\in I_{r_h}}\sum_{j\in J}\sum_{(k\in D_j,k>r_h)}x_{ijk}, \forall h = 0,1,...,t-1. \tag{3.44}$$

Since $\sum_{(k\in D_j,k>r_h)}x_{ijk} \le \sum_{k\in D_j}x_{ijk}$, from equations (3.43) and (3.44), it follows that

$$\sum_{j\in J}\sum_{(k\in D_j,k>r_h)}\lambda_{jk} \le \sum_{i\in I_{r_h}}\sum_{j\in J}\sum_{k\in D_j}x_{ijk}, \forall h = 0,1,...,t-1. \tag{3.45}$$

By constraint (3.5) and (3.45), we note that

$$\sum_{j\in J}\sum_{(k\in D_j,k>r_h)}\lambda_{jk} \le \sum_{i\in I_{r_h}}M_i z_i, \forall h = 0,1,...,t-1, \tag{3.46}$$

from which the result follows. $\square$

Corollary 4. *Constraints (3.38) are valid inequalities for P-MIP.*

Now define Program NMP by replacing Benders' feasibility cuts in the master problem with the feasibility constraints specified by (3.38):

Program NMP

$$\min \sum_{i \in I} f_i z_i + \Theta \tag{3.47}$$

subject to:

$$\Theta \geq \sum_{j \in J} \sum_{(k \in D | \lambda_{jk} > 0)} \lambda_{jk} \mu_{jk} - \sum_{i \in I} M_i \omega_i z_i, (\mu, \eta, \pi, \omega) \in \Omega, \tag{3.48}$$

$$\sum_{i \in I_{r_h}} M_i z_i \geq \sum_{j \in J} \sum_{(k \in D_j, k > r_h)} \lambda_{jk}, \forall h = 0, ..., t-1, \tag{3.49}$$

$$z_i \in \{0,1\}. \tag{3.50}$$

The following proposition shows that Program NMP and Program P-MIP are equivalent.

Proposition 16. *Vector z is feasible for Program P-MIP if, and only if, it is feasible for the Program NMP.*

Proof. The conclusion follows directly from Proposition 3.38 and Lemma 2. □

3.4 RELATIONSHIP AMONG VARIOUS POLYHEDRA

In the previous section, we showed that replacing the set of feasibility cuts in the master problem with our new feasibility constraints (3.38) guarantees the feasibility of the subproblem but does not eliminate any of the integer points feasible in the original problem. In this section we study the efficiency of these constraints by comparing them with Benders' feasibility cuts. We begin by examining how (3.38) relates to the polyhedra defined by the LP relaxation and the convex hull of the original problem.

3.4.1 Eliminating Redundancy

Note that if capacities are large enough that some of the constraints defined by (3.38) can never be satisfied as equalities, then none of these can represent a proper face. In general, it is difficult to come up with an exact characterization of a face or a facet.

Next, we examine these constraints to see whether we can eliminate any redundancy. There are three kinds of dominated constraints in (3.38). A description of these redundancies and how they may be eliminated is as follows.

1) There exist constraints that have the same nonzero coefficients for z variables on the left-hand side, but have different right-hand sides. This happens when there are raw sizes r_h and r_{h+1} such that the subset of facilities with raw materials that are longer than raw size r_h is identical to the subset of facilities with raw materials that are longer than raw size r_{h+1}. In this case, the feasibility constraint (3.38) corresponding to raw size r_{h+1} is always dominated since it has a smaller right-hand side. Therefore, constraint (3.38) for raw size r_{h+1} may be eliminated.

2) There exist constraints that have the same right-hand side, but have different nonzero coefficients for the z variables in the left-hand-side. This could happen when there exists some h, $(h = 0, ..., t-2)$ such that there is no demand for size $k \in (r_h, r_{h+1}]$. In this case, we should eliminate feasibility constraint (3.38) corresponding to raw size r_h since it is always dominated by the constraint for raw size r_{h+1}.

3) There exist constraints with zero right-hand sides. This happens when one or more of the raw sizes are longer than the largest demand size. Suppose the largest demand size k is between raw size $r_{h'-1}$ and raw size $r_{h'}$, we can effectively rewrite the feasibility constraints (3.38) as follows:

$$\sum_{i \in I_{r_h}} M_i z_i \geq \sum_{j \in J} \sum_{(k \in D_j, k > r_h)} \lambda_{jk}, \forall h = 0, 1, ..., h' - 1. \tag{3.51}$$

3.4.2 Definitions

We will now show that our feasibility constraints yield the same set of integer points as Benders' feasibility cuts. This implies that we can obtain the same feasible region as Benders' original procedure but potentially, with far fewer constraints - we only need as many constraints as the number of different raw sizes, while the number of Benders' feasibility cuts required could be much larger. The computational implications of this are also studied in Section 3.5.

Before conducting the polyhedral study, we introduce some notation.

Let N be the total number of variables in Program P-MIP and define Q, the set of feasible points of P-MIP, as follows:

$$Q = \left\{(z, x, y, r, s) \in \mathbb{R}^N \middle| (3.2), (3.3), (3.4), (3.5), x, y, r, s \geq 0, \text{ and } z_i \in \{0,1\}, \forall i \in I\right\}. \tag{3.52}$$

Define the set R of integer 0-1 points that satisfy (3.51) as:

$$R = \left\{ z \in \{0,1\}^{|I|}, \left(\sum_{i \in I_{r_h}} M_i z_i \geq \sum_{j \in J} \sum_{(k \in D_j, k > r_h)} \lambda_{jk}, \forall h = 0, ..., t-1 \right) \right\}. \tag{3.53}$$

and Γ, the set of integer 0-1 points corresponding to Benders' feasibility cuts (3.23) as:

$$\Gamma = \left\{ z \in \{0,1\}^{|I|}, \left(\sum_{i \in I} M_i \omega_i z_i \geq \sum_{j \in J} \sum_{k \in D | \lambda_{jk} > 0} \lambda_{jk} \mu_{jk}, \forall (\mu, \eta, \pi, \omega) \in \Psi \right) \right\}. \tag{3.54}$$

In the next section, we will discuss the relationship between Q, R and Γ.

3.4.3 Relationship among Polyhedra

In this section, we want to show that the convex hull of feasible binary vectors defined by Benders' feasibility cuts is identical to the one defined by non-dominated feasibility constraints (3.51). The first step is to project the binary variables z in vectors belonging to Q onto a subspace $\mathcal{Z}$ where $\mathcal{Z} = \{(z,0)|z \in \mathbb{R}^{|I|}, (z,0) \in \mathbb{R}^N\}$. Let $\mathcal{Z}^\perp = \{(z,x,y,r,s) \in \mathbb{R}^N|z = 0, z \in \mathbb{R}^{|I|}\}$. Let $proj_{\mathcal{Z}}(Q)$ be the projection of Q onto $\mathcal{Z}$, i.e. for any $q = (z,x,y,r,s) \in Q$, $\exists (z',0) \in \mathcal{Z}$ such that $q - (z',0) \in \mathcal{Z}^\perp$. Then we can obtain the following propositions:

Proposition 17. *$proj_{\mathcal{Z}}(Q)$ is nonempty if, and only if, P-MIP is feasible.*

Proof. 1) "$\Rightarrow$" Suppose $proj_{\mathcal{Z}}(Q)$ is nonempty and there exists $(z,0) \in proj_{\mathcal{Z}}(Q)$, then by the definition of projection there exist x,y,r,s such that $q = (z,x,y,r,s) \in Q$. By the definition of Q, it follows that q is feasible in P-MIP.

2) "$\Leftarrow$" Now suppose P-MIP is feasible. Then there exists $q = (z,x,y,r,s) \in Q$. Project q onto the subspace $\mathcal{Z}$. Then there exists $(z',0) \in proj_{\mathcal{Z}}(Q)$ such that $q - (z',0) \in \mathcal{Z}^\perp$. By the definition of $\mathcal{Z}^\perp$, it follows that $z - z' = 0$, i.e. $z = z'$. Thus, $(z,0) \in proj_{\mathcal{Z}}(Q)$. □

Proposition 18. *$R = proj_{\mathcal{Z}}(Q)$.*

Proof. Corollary I.4.12 from (Nemhauser and Wolsey, 1988) states that $R = proj_{\mathcal{Z}}(Q)$ if, and only if, (A) each inequality of the polyhedron R is a valid inequality for Q; (B) $\forall z \in R$, there exists a vector (x,y,r,s) such that $(z,x,y,r,s) \in Q$. Examining both conditions it is clear that:

1) (A) follows directly from Corollary 4.

2) Proposition 15 shows that $\forall z \in R$, Program P-SP(z) is feasible. Since P-MIP has the same additional constraints as P-SP(z), it follows that the former is also feasible. Then from Proposition 17 it follows that $proj_{\mathcal{Z}}(Q)$ is nonempty. Suppose $(z,0) \in proj_{\mathcal{Z}}(Q)$. Then there exists a vector (x,y,r,s) such that $q = (z,x,y,r,s) \in Q$, thus (B) also holds.

The conclusion follows. □

It is well known (Martin, 1999) that Benders' decomposition can be viewed as a projection from the space (z,x,y,r,s) onto $\mathcal{Z}$ space. That is, $\Gamma = proj_{\mathcal{Z}}(Q)$, from which the following corollary follows.

Corollary 5. $\Gamma = R$.

From Corollary 5, we can see that the convex hull determined by Benders' feasibility cuts is the same as the convex hull determined by our feasibility constraints. We present this conclusion as follows:

Corollary 6. *Conv*(Γ) = *Conv*(R).

3.5 COMPUTATIONAL COMPARISONS

In Section 3.4, we showed that our feasibility constraints yield the same set of binary integer points as Benders' feasibility cuts. In practice, our feasibility constraints given by (3.51) can be expected to be much more effective because of the following reasons:

1) The upper bound on the total number of Benders' feasibility cuts to be generated could be an extremely large number, while the upper bound on the number of our feasibility constraints is no more than the number of unique raw sizes.

2) Benders' feasibility cuts could include dominated cuts, while there are no such constraints for our case after we have removed redundancies.

3) At each iteration, Benders' decomposition can generate one and only one feasibility cut, while our constraints are all known *a priori*.

Therefore if our feasibility constraints are all added to the master problem before we begin to solve it, we can expect that at each iteration the improvement in the objective function value will be better than with Benders'. In other words, compared with Benders' feasibility cuts, our feasibility constraints should yield a better lower bound for the master problem (Program MP) at each iteration and therefore reduce the time to reach optimality.

In this section, we compare their effectiveness of our approach through a computational study. We begin by formally defining two algorithms based on our approach that are compared with Benders' decomposition.

3.5.1 Algorithms

The simplest algorithm to avoid Benders' feasibility cuts is to obtain all non-dominated feasibility constraints given by (3.51) and add these to the master problem (thus yielding Program NMP) a priori. We specify this algorithm as follows:

Algorithm 3 Adding all Non-Dominated Feasibility Constraints *a priori*

```
1: Generate all non-dominated feasibility constraints in the set given by (3.51) and add
   them onto the initial restricted master problem (obtained from Program NMP).
2: Solve the restricted master problem and obtain the optimal $^*z$.
3: Solve P-SP($^*z$) and check the stopping criteria; if not satisfied add the corresponding
   optimality cut onto the master problem and go to Step 2.
```

One disadvantage of Algorithm 3 is that the size of the restricted master problem solved at each iteration is relatively large since it has all constraints added initially. A second version of the algorithm (Algorithm 4) adds at each iteration, only those constraints in the set of non-dominated constraints from (3.51) that are needed at that iteration.

Algorithm 4 Adding Feasibility Constraints as Needed

```
1: Generate all non-dominated feasibility constraints in the set given by (3.51); suppose
   there are $b$ of these in all. Define $B = \{1, 2, ..., b\}$, $^*z = \{1, ..., 1\}^{|I|}$ and go to Step 4.
2: Solve the restricted master problem and obtain the optimal vector $^*z$.
3: For each constraint $k \in B$, check whether it is violated by $^*z$; if so set $B = B \backslash k$ and add
   constraint $k$ to the master problem. If any constraint is added to the master problem
   (Program NMP), go to Step 2; otherwise go to Step 4.
4: Solve P-SP($^*z$) and check the stopping criteria; if not satisfied add the corresponding
   optimality cut into the master problem (Program NMP) and go to Step 2.
```

3.5.2 Results

Before exploring the effectiveness of our approach, we report on a study to compare the basic Benders' approach for solving Program P-MIP with a generic MIP solver (we used

Table 10: Performance of Direct and Benders' Approaches

Instance	*Facilities*	*Customers*	*Time (Seconds)*	
			Direct	*Benders*
1	2	2	0.04	0.08
2	2	5	0.10	0.23
3	5	10	2.323	2.323
4	6	12	3.6960	2.583
5	8	30	108.89	41.30
6	10	21	112.19	51.41
7	12	24	177.53	88.93
8	14	17	220.87	118.95
9	17	30	595.96	367.80
10	24	43	2423.41	1663.32
11	27	48	6552.11	4856.34
12	31	53	*	8610.77
13	35	49	*	9474.80
14	38	72	*	10302.32
15	41	78	*	*

Note:"*" indicates that it takes more than (10800 Seconds).

the default solver in CPLEX 7.0. The computers we use in this chapter have Pentium III processor (930MHz) and 256 MB of RAM).

Table 10 indicates that

i) For very small problems (fewer than 5 facilities), it takes longer using Benders' decomposition.

ii) For larger problems the rate of increase in computing time is much higher for the direct approach than Benders' approach. The gap increases as problems become larger.

Table 11: Characteristics of Tested Instances

Instance Set	*No. of Facilities*	*No. of Customers*	*No. of Raw Sizes per Facility*	*No. of Demanded Sizes per Customer*
Small	2 to 4	2 to 5	1 to 5	2 to 10
Medium	5 to 15	6 to 30	2 to 7	5 to 15
Large	16 to 45	31 to 90	3 to 10	8 to 30

iii) The direct method can solve a problem with 27 facilities within a 3-hour time limit; Benders' decomposition can solve a problem that has almost 40 facilities.

In summary, it appears that Bender's Decomposition is an effective alternative to a generic IP solver for nontrivial problem sizes.

The main comparison in this section is between the performance of a traditional Benders' decomposition approach and the two modified versions developed for our problem, namely Algorithms 3 and 4. Our study examined three sets of problems of various sizes, ranging from relatively small to relatively large. Actual instances within each problem set were randomly generated with the parameter values in the ranges shown in Table 11. For each set, 20 problems were randomly generated, and each instance generated was solved independently using each of the three algorithms.

The following performance characteristics are studied for each algorithm:

- TC — The total number of feasibility cuts generated and added to the master problem before reaching optimality.
- DC — The total number of dominated feasibility cuts added to the master problem before reaching optimality.
- AC — The average number of feasibility cuts added to the master problem at each iteration.
- ITE — The total number of iterations to add all feasibility cuts required by the algorithm.

Table 12: Comparison of Three Algorithms

Instance Set	*Item*	*Benders*			*Algorithm 3*			*Algorithm 4*		
		Min.	*Max.*	*Avg.*	*Min.*	*Max.*	*Avg.*	*Min.*	*Max.*	*Avg.*
Small	TC	1	6	2.15	1	5	2.05	1	4	1.45
	DC	0	1	0.15	0	0	0	0	0	0
	AC	1	1	1	N/A	N/A	N/A	1	2	1.20
	ITE	1	6	2.15	N/A	N/A	N/A	1	3	1.15
	TIM	0.05	6.50	1.62	0.03	4.93	1.16	0.03	4.72	1.08
Medium	TC	1	19	7.15	4	12	7.4	1	5	2.75
	DC	0	10	2.95	0	0	0	0	0	0
	AC	1	1	1	N/A	N/A	N/A	1	4	2.1
	ITE	1	19	7.15	N/A	N/A	N/A	1	2	1.35
	TIM	10.24	651.07	150.17	8.15	602.20	128.74	8.00	572.70	120.88
Large	TC	1	27	12.65	10	18	13.32	1	5	3.85
	DC	0	20	6.55	0	0	0	0	0	0
	AC	1	1	1	N/A	N/A	N/A	1	4	2.41
	ITE	1	27	12.65	N/A	N/A	N/A	1	3	1.60
	TIM	362	11040	3752	331	4838	3013	259	4454	2806

Note:

"N/A": Since Algorithm 3 adds all feasibility constraints to the master problem before it starts to solve the subproblems AC and ITE are not applicable.

- TIM — The total time (in seconds) to solve the problem.

The complete computational results can be found Chapter B in Appendix. They are summarized in Table 12. These results indicate the following:

i) In general, when the problem becomes larger more feasibility cuts/constraints are required to find the optimum. However, using Algorithm 4 results in significantly fewer cuts being added when compared to Benders' original approach and Algorithm 3.

ii) Even though Algorithm 3 adds all constraints initially, the approach used in Algorithm 4 of adding feasibility constraints as required seems to be more efficient. In every instance, Algorithm 4 outperforms Algorithm 3; while the total number of iterations required to reach optimality with Algorithm 4 might be a little larger the time spent at each iteration is lower because of the smaller master problems solved each time. The total savings in computational time average about 7%.

iii) On average, Algorithm 4 adds more than one feasibility constraint to the master problem (Program NMP) at each iteration; Benders' approach always adds exactly one. This indicates that our approach tightens the feasible region more quickly.

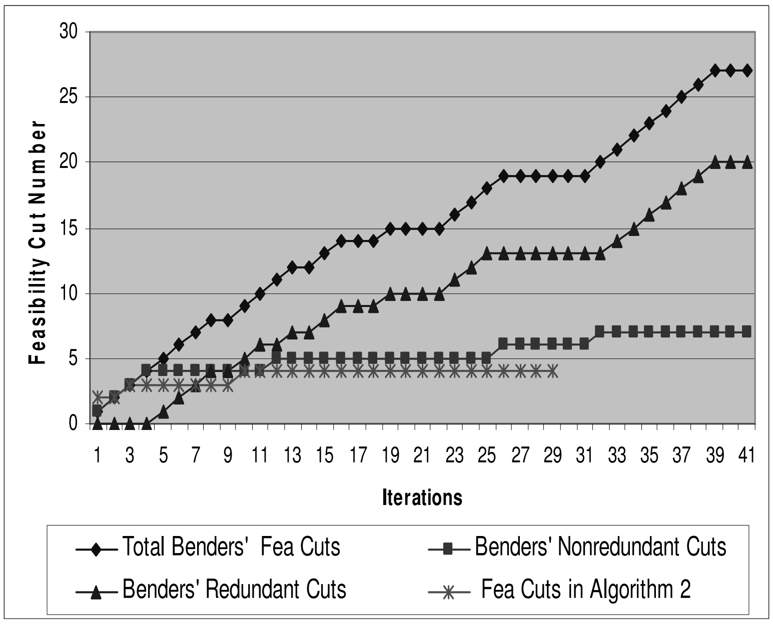

Figure 7: Redundancy of Benders' Feasibility Cuts

iv) Algorithms 3 and 4 both result in no dominated feasibility constraints. However, with Benders' approach feasibility cuts generated at earlier iterations are often rendered dominated by the cuts generated later. This is particularly true for the larger problem instances. Figure 7 displays results for one large problem where 20 of 27 feasibility constraints (74%) generated by Benders' procedure are dominated at the optimum, while Algorithm 4 only needs 4 feasibility constraints. This indicates that for this problem our feasibility constraints are more efficient than the traditional Benders' cuts.

v) Compared with Benders' decomposition it takes both Algorithms 3 and 4 less time to solve the problem. Table 12 shows that compared with Algorithm 3, on average Benders' decomposition takes between about 17% and 40% longer to solve the problems in our test sets; compared with Algorithm 4 the corresponding values range from about 24% to 50%.

vi) The results in Table 12 support the conclusion that our approach is also more efficient at each iteration. The total number of iterations to add all required feasibility constraints for Algorithm 4 is significantly lower than Benders' approach: for the large problems, it takes Benders' decomposition over 12 iterations on average, while the corresponding number for Algorithm 4 is under two.

Table 13: Comparison between Benders and Algorithm 4

Instance	*Facilities*	*Customers*	*Time (Seconds)*	
			Benders	*Algorithm 4*
1	31	44	7812	5085
2	35	46	8911	5832
3	38	60	9823	6776
4	40	64	*	7928
5	43	69	*	8690
6	46	63	*	9437
7	51	61	*	10682
8	56	76	*	*

Note: "*" indicates that it takes more than 10800 Seconds.

Finally, we compared Benders' decomposition with Algorithm 4 to find the largest problem that each one could solve within a 3-hour time limit. The results in Table 13 indicate that within this time limit Algorithm 4 could solve a problem with as many as 51 facilities, while Benders' approach could not solve problems with more than 40 facilities.

4.0 STOCHASTIC MODEL

4.1 BACKGROUND

In the previous section, we considered a mixed-integer program in which we assumed that parameters such as the shipping costs and demand rates were known a priori. In practice, this may not be true. Whereas deterministic optimization problems are formulated with known parameters, real world problems almost invariably include some unknown parameters. Some parameters might be random variables with values that depend on random events. In other words, the decision on which facilities to open might have to be taken without full information on some random events. Later, full information is received and corrective production and distribution actions are taken. Therefore, we may need to consider an optimization problem with uncertain information.

To model this stochastic and dynamic environment, we will utilize stochastic programming [11, 39, 58]. Stochastic programs are mathematical programs where some of the data incorporated into the objective or constraints are uncertain. Uncertainty is usually characterized by a probability distribution on the parameters. We assume the distribution of this uncertainty is discrete and each element in the distribution is defined as a *scenario.* The outcomes are generally described in terms of scenarios. For example, they can be the set of possible demands over the next few months.

In a classical stochastic program, we make one decision now and minimize the expected costs of the consequences of that decision. The most widely applied and studied stochastic programming models are two-stage programs. In these programs, the decision maker takes some action in the first stage, after which a random event occurs affecting the outcome of the first-stage decision. A recourse decision can then be made in the second stage that depends on the first-stage decision as well as the random outcome. The optimal policy from such a model is a single first-stage decision and a collection of recourse decisions defining which second-stage action should be taken in response to each random outcome.

When the random variable is discretely distributed, the problem can be written as a large deterministic problem. The expectations can be written as finite sums, and each constraint can be duplicated for each realization of the random data. While in theory the resulting extensive form can be solved using any general purpose optimization package, in practice, it is too large to do so. For computational methods in stochastic programming, Wollmer [114] introduces a two-stage stochastic program with a binary first stage. Subsequently, several heuristics were developed to solve the problem [61, 67, 76, 116], but several computational issues still need to be resolved. Holmberg and Tuy [54] consider a production-distribution problem where the demand is stochastic and there is a convex penalty for unmet demand, and develop a branch and bound procedure to solve the reduced problem. Louveaux and Peeters [73] develop a dual-based heuristic to solve the uncapacitated stochastic facility location problem. Wang et al. [111] consider a facility location problem with stochastic demand to decide the locations of immobile servers and develop three heuristics based on the greedy dropping procedure, tabu search approach, and branch-and-bound method to solve the problem. Louveaux and Hamme [72] develop a branch-and-bound method to solve a capacitated facility location problem optimally, but they do not consider the production problem and their method could only solve problems with up to 10 potential facility locations and 40 customers. A survey of heuristics and algorithms for stochastic location problems can be found in Louveaux [71].

These extensive form models often have a block-angular structure [13]. Also, several models are MIP problems, i.e., problems with binary variables, which makes the problem hard to solve. One of the most popular methods for obtaining optimal solutions to stochastic programs is the L-shaped method [107], which follows an approach similar to Benders' decomposition [74] in order to separate the binary variables and to decompose the problem by exploiting the block angular structure.

Suppose ξ describes a possible realization, or *scenario*, and that ξ can take on a finite number of different values, each with some known probability. Let Ξ be this finite support for the random variable ξ. Assume f_1, f_2 and z_1, z_2 are vectors of cost coefficients and variables, respectively, where the values of f_2 and z_2 depend on the particular scenario that is realized. We can write the standard extensive form for stochastic program with binary variables as follows:

Extensive Form

$$\min {f_1}^T z_1 + E_\xi \left[(f_2(\xi))^T z_2(\xi) \right] \tag{4.1}$$

subject to:

$$A z_1 = b, \tag{4.2}$$

$$T(\xi) z_1 + W z_2(\xi) = h(\xi), \xi \in \Xi, \tag{4.3}$$

$$z_1 \in \mathbb{B}^n, z_2(\xi) \geq 0, \xi \in \Xi. \tag{4.4}$$

Note that A and W are fixed matrices and b is a fixed vector, while $T(\xi)$ and $h(\xi)$ depend on the scenario ξ. To exploit block-angular structure in the above extensive form, the L-shaped method decomposes the primal problem into a master problem and many subproblems. Like Benders' decomposition, the L-shaped method first solves the master problem to obtain a solution for the vector z_1, and then passes this to each subproblem. If it is infeasible for some subproblem, a feasibility cut is added onto the master problem; otherwise optimality cuts are added. Depending on how many optimality cuts are added onto the master problem at each iteration, the L-shaped method has two versions: the single-cut version and the multi-cut version [11]. We briefly present these two methods in the following sections.

4.1.1 The Single-Cut L-shaped Method

The motivation behind the single-cut L-shaped method is that large stochastic problems usually have hundreds or even thousands of scenarios. Adding an optimality cut based on the dual values for every scenario could lead to many constraints after each iteration, which causes the master problem for the single-cut L-shaped method to rapidly become very large. This weakens the efficiency of the L-shaped method very quickly since the master problem for the single-cut L-shaped method is a mixed-integer program and is, in general, hard to solve.

Considering this difficulty, at each iteration, the single-cut version solves all second-stage optimality subproblems and combines all their dual information together to only generate one optimality cut and add it to the master problem. Therefore, for the single-cut version, the size of the master problem increases by no more than one optimality cut at each iteration, which make it less sensitive to the increasing number of scenarios.

Let θ represent the expected recourse function $\mathcal{Q}(z)$. The details of the L-shaped algorithm [39, 93] are as follows:

Step 0. Set $w_1 = w_2 = \nu = 0$.

Step 1. Set $\nu = \nu + 1$, solve the following master problem and let (z_1^ν, θ^ν) be its optimal solution.

<u>Master Problem for the Single-Cut L-shaped Method</u>

$$\min \ {f_1}^T z_1 + \theta \tag{4.5}$$

subject to:

$$W_{l_1} z_1 \geq d_{l_1}, l_1 = 1, ..., w_1, \tag{4.6}$$

$$E_{l_2} z_1 + \theta \geq e_{l_2}, l_2 = 1, ..., w_2, \tag{4.7}$$

$$\theta \in \Re, z_1 \in \mathbb{B}^n. \tag{4.8}$$

Let (z_1^ν, θ^ν) be its optimal solution. Constraints (4.6) are called *feasibility cuts*, which are generated from the feasibility subproblem described in Step 2. Constraints (4.7) are *optimality cuts*, which are generated from the optimality subproblem described in Step 3.

Step 2. Obtain the feasibility cut. For every $\xi = 1, 2, ..., |\Xi|$, we solve the following feasibility subprogram

<u>Feasibility Subproblem</u>

$$\min \ \ \omega' = e^T v^+ + e^T v^- \tag{4.9}$$

subject to:

$$W z_2 + I v^+ - I v^- = h(\xi) - T(\xi) z_1^\nu, \tag{4.10}$$

$$z_2 \geq 0, v^+ \geq 0, v^- \geq 0. \tag{4.11}$$

where $e^T = (1, 1, ..., 1)$.

If the optimal value ω' for that scenario is positive, define ϱ^{ν} as dual variable associated with (4.10). Set $w_1 = w_1 + 1$, generate the feasibility cut (4.6) using $W_{w_1} = (\varrho^{\nu})^T T(\xi)$ and $d_{w_1} = (\varrho^{\nu})^T h(\xi)$, and add it to the master problem. If no feasibility cuts are added at this step, i.e., $\omega' \leq 0$ for every scenario ξ, then go to Step 3, otherwise go to Step 1.

Step 3. Solve the following optimality subproblem for $\xi = 1, ..., |\Xi|$.

Optimality Subproblem

$$\min \ \ \omega = (f_2(\xi))^T z_2 \tag{4.12}$$

subject to:

$$Wz_2 = h(\xi) - T(\xi)z_1^{\nu}, \tag{4.13}$$

$$z_2 \geq 0. \tag{4.14}$$

Suppose π_{ξ}^{ν} represents the associated optimal dual values for the optimality subproblem under scenario ξ. Define $E_{w_2} = \sum_{\xi=1}^{|\Xi|} p(\xi)(\pi_{\xi}^{\nu})^T T(\xi)$ and $e_{w_2} = \sum_{\xi=1}^{|\Xi|} p(\xi)(\pi_{\xi}^{\nu})^T h(\xi)$, where $p(\xi)$ is the probability of scenario ξ.

Let $\omega^{\nu} = e_{w_2} - E_{w_2} z_1^{\nu}$. If $\theta^{\nu} \geq \omega^{\nu}$, stop; $(z_1^{\nu}, \theta^{\nu})$ is an optimal solution. Otherwise set $w_2 = w_2 + 1$ and add the constraint (4.7) to the master problem defined in Step 1. Return to Step 1.

4.1.2 The Multi-Cut L-shaped Method

Interestingly, the shortcomings of the single-cut L-shaped method arise from combining the dual information from optimal multipliers for all scenarios to generate only one cut, because this weakens the strength of the cut. The reason is that this aggregation ignores any special information for a specific scenario. Thus, its convergence might be slow and it could take many iterations to solve the problem. In 1988 Birge and Louveaux [12] presented a new multi-cut L-shaped method and showed that the total number of iterations for the multi-cut version is less than or equal to the total number for the single-cut version. In this new method, instead of generating one optimality cut using all dual information for all scenarios, it emphasizes the dual values for each scenario by generating a cut based on the optimal multipliers of each scenario.

In this section, we discuss this new method in more detail. The feasibility cuts in the multi-cut L-shaped method are the same as the ones in the single-cut version. The main difference is at Step 3. In the multi-cut version, we may add an optimality cut for *each* scenario and thus as many as $|\Xi|$ optimality cuts to the master problem per iteration. Let θ_ξ represent the expected recourse function under scenario ξ, $Q(z_1, \xi)$. We can rewrite the new master problem with multi-cut as follows:

Master Problem for the Multi-Cut L-Shaped Method

$$\min \ {f_1}^T z_1 + \sum_{\xi=1}^{|\Xi|} \theta_\xi \tag{4.15}$$

subject to:

$$W_{l_1} z_1 \geq d_{l_1}, l_1 = 1, ..., w_1, \tag{4.16}$$

$$E_{l_2(\xi)} z_1 + \theta_\xi \geq e_{l_2(\xi)}, l_2(\xi) = 1, ..., w_2(\xi), \xi = 1, ..., |\Xi|, \tag{4.17}$$

$$z_1 \in \mathbb{B}, \theta_\xi \in \Re, \forall \xi = 1, ..., |\Xi|. \tag{4.18}$$

The procedure of the multi-cut L-shaped method is as follows:

Step 0. Set $w_1 = \nu = 0$ and $w_2(\xi) = 0, \forall \xi = 1, ..., |\Xi|$.

Step 1. Set $\nu = \nu + 1$, solve the master problem for the multi-cut version and let $(z_1^\nu, \theta_1^\nu, ..., \theta_{|\Xi|}^\nu)$ be its optimal solution.

Step 2. As before in the single-cut L-shaped method.

Step 3. For $\xi = 1, ..., |\Xi|$, solve the linear optimality subproblem described in Step 3 for the single-cut L-shaped method. Suppose π_ξ^ν represents the optimal dual variables for scenario ξ. If

$$\theta_\xi^\nu < p(\xi)(\pi_\xi^\nu)^T(h(\xi) - T(\xi)z_1^\nu), \tag{4.19}$$

define: $E_{l_2(\xi)+1} = p(\xi)(\pi_\xi^\nu)^T T(\xi)$ and $e_{l_2(\xi)+1} = p(\xi)(\pi_\xi^\nu)^T h(\xi)$. Set $l_2(\xi) = l_2(\xi) + 1$ and add an optimality cut corresponding to (4.17). If (4.19) does not hold for any $\xi = 1, ...|\Xi|$, stop; z_1^ν is an optimal solution. If any optimality cuts have been added to the master problem, return to Step 1.

4.1.3 Overview of Our Approach

In the next section of this chapter, we first develop a two-stage stochastic model to describe our problem. Then, to decide which facilities to open, we use the L-shaped method [11, 94] to solve the problem. In Chapter 3, we have shown that even the deterministic version of this problem is NP-hard. Therefore, the stochastic program developed in this chapter is clearly NP-hard. It is very time-consuming to solve this two-stage stochastic model even using a solution approach such as the L-shaped method, which could theoretically solve the second-stage problems an exponentially large number of times.

In this research, we try to enhance the performance of the traditional L-shaped method along the following two directions:

1) Instead of generating either one or multiple optimality cuts using all dual information under all scenarios, we develop a new algorithm, which we call the double-cut algorithm to generate two optimality cuts using only selected dual information at each iteration. This appears to increase the efficiency of the optimality cuts.
2) Instead of solving the second-stage problem to generate feasibility cuts, we generate constraints without solving the feasibility problems and add these to the first-stage problem so as to guarantee the feasibility of the second-stage problem. By doing so, we eliminate the step of solving the feasibility problem in the L-shaped method. We also show that our feasibility constraints are more effective because the upper bound on the number of feasibility constraints we generate is polynomial in the size of the instance.

Based on the above two improvements, we develop new algorithms to solve our stochastic program. To check the effectiveness of these new algorithms, we also perform computational tests on randomly generated test problems.

4.2 FORMULATION

In this section, we handle a more general case of our problem by extending the version described in Chapter 3 and restating it as a stochastic program. In this new model, all parameters described before, except the fixed facility cost, are considered uncertain and have different realization values under different random scenarios. These uncertain parameters as treated as random variables. An accurate probabilistic description of these random variables is assumed available under some form of discrete probability distribution. The particular values these random variables will assume are only known after a specific scenario is realized. In this new model, the set of decisions is divided into two groups:

1) The *first-stage decisions* on which facilities to open has to be taken before we know the values of these random variables.
2) The *second-stage decisions* on how to cut raws / remnants and how to distribute the products to satisfy all demand are taken after we know the values of these random variables.

Compared with the problem described in Chapter 3, the one described in this section is complicated by the fact that the decision on which facilities to open must be made without full information. The decisions on which sizes to cut and how to distribute the product are also complicated by the fact that parameters such as the cutting costs, raw costs, demand rates, etc., typically have different values under different random scenarios. The goal is to minimize the sum of the fixed costs, the expected material costs, cutting costs and transportation costs, minus the expected recycled scrap value.

Suppose that the λ's are discretely distributed, and ξ describes a possible realization, or *scenario*. Suppose ξ can take on a finite number of different values and let Ξ be this finite support for the random variable ξ. In this chapter, we first consider a model in which in addition to the demand rates, the cutting costs, scrap value, facility capacity, the sizes, and costs of raw materials at each facility $i \in I$, the demand types and sizes arising from customer $j \in J$, and the shipping costs from the facilities to the customers are all permitted to be stochastic. Initially, we assume that all parameters could be stochastic and develop a corresponding model. Then, we present a model for the case where the capacity and the stock raw sizes at each facility do not change under different scenarios $\xi \in \Xi$.

To solve the problem, we must first decide on a set of facility locations to open without full information, so that it minimizes the sum of the discounted fixed costs and the expected future operating costs and transportation costs, minus the expected recycled scrap value. After this decision has been made, we then fix the binary location variables and solve the second-stage recourse linear programs. By solving the perturbed recourse LP model, as described in Chapter 2, we can obtain the inherent value for each size at each open facility under each scenario. Based on these inherent values, we can then apply the same inherent value policy we developed in Chapter 3 to manage the long-run costs under dynamic demand.

4.2.1 Model

To describe our model, we first introduce some notation. Sets I and J are defined as before:

- I = Set of facility locations.
- J = Set of demand locations.

We introduce some new notation and modifications to our prior notation as follows:

- $\xi \in \Xi$ = Possible scenario, where Ξ is the set of all scenarios and has a discrete distribution.
- $p(\xi)$ = Probability of scenario ξ.
- $K_i(\xi)$ = Set of all possible sizes, including scrap sizes, that could be generated at facility $i \in I$ under scenario ξ (we include the size 0 in this set); define $K = \bigcup_{i \in I, \xi \in \Xi} K_i(\xi)$.
- $D_j(\xi)$ = Set of all sizes that are demanded at demand location $j \in J$ under scenario ξ; define $D = \bigcup_{j \in J, \xi \in \Xi} D_j(\xi)$.
- $S_i(\xi)$ = Set of all raw stock sizes that are processed at facility location $i \in I$ under scenario ξ; define $S = \bigcup_{i \in I, \xi \in \Xi} S_i(\xi)$.
- $\lambda_{jk}(\xi)$ = Demand rate at location $j \in J$ for size $k \in D_j(\xi)$ under scenario ξ.
- $c_{ijk}(\xi)$ = Cost of transporting a single unit of size $k \in D_j(\xi)$ from facility location $i \in I$ to demand location $j \in J$ under scenario $\xi \in \Xi$.
- $a_{ih}(\xi)$ = Cost of one unit of raw stock of size $h \in S_i(\xi)$ at facility location $i \in I$ under scenario $\xi \in \Xi$.
- $\sigma_i(\xi)$ = Unit salvage value of product at facility $i \in I$ under scenario $\xi \in \Xi$; $\sigma_i > 0$.
- $\delta_i(\xi)$ = Cost per cut at facility $i \in I$ under scenario $\xi \in \Xi$; $\delta_i > 0$.

- $M_i(\xi)$ = Total open production capacity for facility $i \in I$ per unit time under scenario $\xi \in \Xi$.
- f_i= Equivalent uniform daily fixed cost [84] of operating a facility at location $i \in I$.

The decision variables are:

- z_i = Variables to decide whether facility i is open or not, if facility $i \in I$ is open, $z_i = 1$; otherwise, $z_i = 0$.
- $x_{ijk}(\xi)$ = Rate of shipment of units of size $k \in D_j(\xi)$ from facility location $i \in I$ to demand location $j \in J$ under scenario $\xi \in \Xi$.
- $y^i_{m,n}(\xi)$ = Rate of generation at facility location $i \in I$ of units of size $(m-n)$ that are obtained by cutting size m down to size n under scenario $\xi \in \Xi$.
- $r_{ih}(\xi)$ = Rate of replenishment of raw units of size $h \in S_i(\xi)$ at facility location $i \in I$ under scenario $\xi \in \Xi$; define $r_{ih}(\xi) = 0$ if $h \in (K_i(\xi) \backslash S_i(\xi))$ under scenario $\xi \in \Xi$.
- $s_{iu}(\xi)$ = Rate at which size u is scrapped at facility location $i \in I$; define $s_{iu}(\xi) = 0$ if the size u cannot be generated at location i under scenario $\xi \in \Xi$.

Before formulating the problem, we also define the sub-network under scenario ξ, as $G_i(\xi) = (K_i(\xi), A_i(\xi))$, where $K_i(\xi)$ is the set of nodes and $A_i(\xi)$ is the set of arcs under scenario ξ in the graph.

We are now ready to formulate the problem as the following stochastic program (SP):

<u>**Extensive Form of SP Model (Program SP)**</u>

$$\min \sum_{i \in I} f_i z_i + E_\xi \Bigg[\sum_{i \in I} \sum_{h \in S} a_{ih}(\xi) r_{ih}(\xi) + \sum_{i \in I} \sum_{j \in J} \sum_{k \in D} c_{ijk}(\xi) x_{ijk}(\xi)$$

$$+ \sum_{i \in I} \sum_{(m,n) \in A_i(\xi) | n > 0} \delta_i(\xi) y^i_{m,n}(\xi) - \sum_{i \in I} \sum_{u \in K} u \sigma_i(\xi) s_{iu}(\xi) \Bigg] \tag{4.20}$$

subject to:

$$\left(r_{il}(\xi) + \sum_{m | (m,l) \in A_i(\xi)} y^i_{m,l}(\xi) \right) - \left(s_{il}(\xi) + \sum_{n | (l,n) \in A_i(\xi)} y^i_{l,n}(\xi) \right) = 0,$$

$$\text{for all } i \in I,\ l \in K_i(\xi), \xi \in \Xi, \tag{4.21}$$

$$\sum_{i \in I} x_{ijk}(\xi) = \lambda_{jk}(\xi), \text{ for all } j \in J, \{k \in D | \lambda_{jk} > 0\}, \xi \in \Xi, \tag{4.22}$$

$$\left(\sum_{(m,n) \in A_i(\xi) | (m-n)=k} y^i_{m,n}(\xi) \right) - \left(\sum_{j \in J} x_{ijk}(\xi) \right) = 0, \text{ for all } i \in I, k \in D, \xi \in \Xi, \tag{4.23}$$

$$\sum_{j \in J} \sum_{k \in D} x_{ijk}(\xi) \leq M_i(\xi) z_i, \text{ for all } i \in I, \xi \in \Xi, \tag{4.24}$$

$$z_i \in \{0,1\}; x(\xi), y(\xi), r(\xi), s(\xi) \geq 0, \text{ for all } \xi \in \Xi. \tag{4.25}$$

It is not difficult to see that the objective function (4.20) is to minimize the fixed costs of facilities, plus the expected costs of raw materials, cutting and transportation, minus the value of the scrap that is salvaged, over all scenarios.

4.2.2 Solution Procedure

The problem described in Section 4.2.1 is a two-stage stochastic program with a binary first stage. Our approach will be used to solve the stochastic program and obtain the inherent values for all sizes at all facilities. We apply the L-shaped method to solve Program and obtain the dual values from its linear subproblem. The policy that was outlined in Section 3.1.2 will then be used to manage long-run costs under dynamic demand.

In the L-shaped method, the first-stage corresponds to the facility location problem, and the second-stage is the operational phase. Note that the master problem only has binary variables and each subproblem, or *recourse problems*, is defined for one scenario and uses parameter values realized under that scenario. We first present the first-stage master problem as follows:

Master Problem in SP

$$\min \sum_{i \in I} f_i z_i + \mathcal{Q}(z) \tag{4.26}$$

subject to:

$$z_i \in \{0,1\}, i \in I. \tag{4.27}$$

Let $Q(z, \xi)$ be the optimal value of second-stage problem for scenario ξ in the stochastic program, which is defined below in (4.28). Then $\mathcal{Q}(z)$, the *expected recourse function*, is given by $\mathcal{Q}(z) = \sum_{\xi=1}^{|\Xi|} p(\xi) Q(z, \xi)$. The expected value function of the problem is continuous and convex. For each scenario, there is a corresponding recourse problem. Each recourse problem is analogous to the Benders' subproblem described in Chapter 3. Once the random vector z has been realized, the recourse problems will be solved. After solving the recourse problems, two types of constraints are sequentially added: 1) feasibility cuts determining the vector z feasible for all recourse problem, 2) optimality cuts, which are supporting hyperplanes and linear approximations to the expected recourse function on its domain. The dual values of these second-stage problems will then be used as the η variables in the proposed policy for the dynamic environment. The recourse problem are defined as follows:

The Second-Stage Problem in SP (Program SSP)

$$Q(z,\xi) = \min \sum_{i\in I}\sum_{j\in J}\sum_{k\in D} c_{ijk}(\xi)x_{ijk} + \sum_{i\in I}\sum_{h\in S} a_{ih}(\xi)r_{ih}$$

$$+\sum_{i\in I}\sum_{(m,n)\in A_i(\xi)|n>0} \delta_i(\xi)y^i_{m,n} - \sum_{i\in I}\sum_{u\in K} u\sigma_i(\xi)s_{iu} \tag{4.28}$$

subject to:

$$\left(r_{il} + \sum_{m|(m,l)\in A_i(\xi)} y^i_{m,l}\right) - \left(s_{il} + \sum_{n|(l,n)\in A_i(\xi)} y^i_{l,n}\right) = 0,$$

$$\text{for all } i\in I \text{ and } l\in K_i(\xi), \tag{4.29}$$

$$\sum_{i\in I} x_{ijk} = \lambda_{jk}(\xi), \text{ for all } \ j\in J \text{ and } k\in D, \tag{4.30}$$

$$\left(\sum_{(m,n)\in A_i(\xi)|(m-n)=k} y^i_{m,n}\right) - \left(\sum_{j\in J} x_{ijk}\right) = 0, \text{ for all } i\in I \text{ and } k\in D, \tag{4.31}$$

$$\sum_{j\in J}\sum_{k\in D} x_{ijk} \le M_i(\xi)z_i, \text{ for all } i\in I, \tag{4.32}$$

$$x,y,r,s \ge 0. \tag{4.33}$$

For our model, the above second-stage problem is the optimality subproblem. Before the second-stage optimality problem is solved, we need to know whether it is feasible for the optimal vector z that we obtain from the master problem.

Recall that in the L-shaped method described in Section 4.1.1, before solving Program SSP for a given vector z in Step 3, we solve a feasibility subproblem in Step 2 to ascertain second-stage feasibility for vector z. If this is infeasible, we generate a feasibility cut based on dual information from this problem and add it into the master problem to reduce the feasible region for the vector z. We will keep adding these feasibility cuts until we obtain a vector z that is feasible for all optimality subproblems. Feasibility cuts are generated from the extreme rays of the dual program of the above program SSP to restrict the vector z in the master problem. This work is done by solving the following feasibility subproblem (Program SFP):

Feasibility Subproblem in SP (Program SFP)

$$\min \ \omega' = \sum_{i\in I}\sum_{l\in K_i} v^+_{1il} + \sum_{j\in J}\sum_{k\in D} v^+_{2jk} + \sum_{i\in I}\sum_{k\in D} v^+_{3ik} + \sum_{i\in I}\sum_{l\in K_i} v^-_{1il} + \sum_{j\in J}\sum_{k\in D} v^-_{2jk} + \sum_{i\in I}\sum_{k\in D} v^-_{3ik} + \sum_{i\in I} v^-_{4i} \tag{4.34}$$

subject to:

$$\left(r_{il} + \sum_{m|(m,l)\in A_i(\xi)} y^i_{m,l}\right) - \left(s_{il} + \sum_{n|(l,n)\in A_i(\xi)} y^i_{l,n}\right) + v^+_{1il} - v^-_{1il} = 0, \text{for all } i \in I \text{ and } l \in K_i, \tag{4.35}$$

$$\sum_{i\in I} x_{ijk} + v^+_{2jk} - v^-_{2jk} = \lambda_{jk}(\xi), \text{ for all } \ j \in J \text{ and } k \in D, \tag{4.36}$$

$$\left(\sum_{(m,n)\in A_i(\xi)|(m-n)=k} y^i_{m,n}\right) - \left(\sum_{j\in J} x_{ijk}\right) + v^+_{3ik} - v^-_{3ik} = 0, \text{ for all } i \in I \text{ and } k \in D, \tag{4.37}$$

$$\sum_{j\in J}\sum_{k\in D} x_{ijk} - v^-_{4i} \le M_i(\xi) z_i, \text{ for all } i \in I, \tag{4.38}$$

$$x, y, r, s, v^+, v^- \ge 0, \tag{4.39}$$

where $v^+ = (v^+_{1i}, v^+_{2j}, v^+_{3i})$ and $v^- = (v^-_{1i}, v^-_{2j}, v^-_{3i}, v^-_{4})$.

If a vector z is feasible for all optimality subproblems, i.e. $\omega' = 0$ for all scenarios, we will check the optimality of this vector z. If it is not optimal for Program SP, then we need to generate optimality cuts and add them into the master problem.

4.2.3 Comparison of Single and Multi-Cut Methods

The single-cut and multi-cut methods both have their advantages and disadvantages [11, 89, 94]. The advantage of the multi-cut version is that unlike the single-cut version we can make good use of dual information from each scenario. Its disadvantage is that if there are many scenarios or if the number of iterations to solve a problem is large, we may add too many optimality cuts, potentially taking much longer to solve the problem. Gassmann [39] concludes that for large problems, the single-cut version generally performs better than the multi-cut version. This conclusion is especially true for a problem such as ours with a mixed-integer first-stage problem, because compared with a linear program, a mixed-integer program is much harder to solve and much more sensitive to the size of the problem. Therefore, the single-cut version is likely to be be more suitable to solve large-scale problems for our SP model. This general observation is supported by the results from a computational study that was conducted. The results are shown in Table 14, where the instances are listed in increasing order of size. The computers we use in this chapter have Pentium 4 processor (2.40GHz) and 2.00 GB of RAM. CPLEX is version 7.0.

Table 14 shows that for large problems, although the multi-cut L-shaped method takes fewer iterations to solve the problem, it adds many more optimality cuts into the first-stage problems and generates much larger mixed-integer problems. Therefore, in terms of the total time to obtain the optimal solutions, it is dominated by the single-cut version for these large problems. We therefore choose the single-cut L-shaped method to solve our stochastic problem.

In the following sections, we only focus on the single-cut L-shaped method. We explore how to improve the performance of the single-cut version by presenting a new method to generate optimality cuts based on selected dual information from the second-stage problem. We also enhance its performance by developing a new method to generate a polynomially bounded number of new constraints to replace the feasibility cuts, so as to ensure the feasibility of the second-stage optimality problems under all scenarios.

Table 14: Performance of Single and Multi-cut L-shaped Methods

Instance	*Single-cut*		*Multi-cut*		
	Iterations	Time (Seconds)	# of Optimality Cuts	Iterations	Time (Seconds)
1	4	12.705	31	3	11.27
2	8	257.88	166	6	249.55
4	3	281.4	227	3	382.2
3	4	294.59	178	4	448.835
5	5	410.34	304	4	473.67
6	5	553.98	611	5	632.85
7	16	581.35	38	9	790.91
8	3	657.09	389	3	871.34
10	7	269.04	39	6	1044.37
9	4	1527.61	436	4	1877.19
11	9	2194.85	1147	7	3189.22
12	10	2724.47	607	7	3310.58
13	8	2974.44	806	7	3885.49
14	18	3167.49	78	12	4484.62
15	8	3997.91	647	6	4780.65
16	16	5146.31	1565	13	7411.78
Avg	8.00	1565.72	454.31	5.75	2115.28

4.3 THE DOUBLE CUT L-SHAPED METHOD

As discussed in the previous section, the L-shaped method with single-cut is better for solving large-scale problems, especially for problems with a mixed-integer first-stage problem. However, simply aggregating all dual information from the second-stage problems might weaken the cut.

Let us consider a stochastic program whose second-stage optimality problem is a minimization problem. For a minimization problem, its primal objective function value is an upper bound on its optimal objective value, while its dual objective function value is a lower bound. Therefore, the primal objective function value $Q(z, \xi)$ (or θ_o) must be larger than or equal to the dual objective function value $e_l - E_l z$. In the L-shaped method, we obtain the recourse function value, θ_o, from the master problem. Then, when we solve the second-stage problem SSP for each scenario ξ, we check whether the above criterion has been met. If it is not satisfied, θ_o obtained in the master problem is not optimal for this scenario ξ and we need to add an optimality cut into the master problem to increase θ_o. If the lower bound of the second-stage problem is equal to its upper bound, then the solution to the second-stage problem corresponding to this scenario is optimal and we do not need to increase θ_o. Therefore, the optimality cut based on these optimal scenarios is weak. Combining these scenarios with those scenarios that are not optimal will weaken the cut, and could slow down convergence and result in more iterations to solve the problem.

In this section, we first develop a new method to quantify this phenomenon and separate these two different groups of scenarios: one whose subproblem is optimal and the other suboptimal. We then generate two cuts based on the dual information from the scenarios in these two groups, respectively.

Before introducing this new algorithm, first let us define a vector γ_{l1}. Define $\gamma_{(l1,o)}(o = 1, 2, ...|\Xi|)$ as an indicator such that it is equal to 1 if the second-stage optimality problem for scenario o at iteration l is not optimal; 0 otherwise. Then we can introduce the master problem for the double-cut L-shaped method as follows:

The Double Cut Master Problem

$$\min \sum_{i\in I} f_i z_i + \sum_{o=1}^{|\Xi|} \theta_o \tag{4.40}$$

subject to:

$$W_l z \geq d_l, l = 1, ..., r, \tag{4.41}$$

$$E_{l_1} z + \sum_{o=1}^{|\Xi|} \gamma_{(l_1,o)} \theta_o \geq e_{l_1}, l_1 = 1, ..., q_1, \tag{4.42}$$

$$E_{l_2} z + \sum_{o=1}^{|\Xi|} (1 - \gamma_{(l_1,o)}) \theta_o \geq e_{l_2}, l_2 = 1, ..., q_2, \tag{4.43}$$

$$\forall z_i \in \{0,1\}, \forall i \in I, \theta_o \in \Re, o = 1, ..., |\Xi|. \tag{4.44}$$

We refer to the optimality cut (4.42) as the aggregated violated optimality cut since each θ_o in the set is not optimal and the cut based on these scenarios will be active and increase their values (θ_o) in the master problem at the next iteration. Correspondingly, we refer to cut (4.43) as the aggregated nonviolated optimality cut.

Now, we are ready to present the procedure for the double-cut L-shaped method as follows:

Step 0. Set $w = \nu = q_1 = q_2 = 0$.

Step 1. Set $\nu = \nu + 1$, solve the master problem for the double cut method and let $(z^\nu, \theta_1^\nu, ..., \theta_{|\Xi|}^\nu)$ be its optimal solution.

Step 2. For every $\xi = 1, 2, ..., |\Xi|$, we solve the following feasibility subprogram

Feasibility Subproblem

$$\min \ \ \omega' = e^T \upsilon^+ + e^T \upsilon^- \tag{4.45}$$

subject to:

$$W z_2 + I\upsilon^+ - I\upsilon^- = h(\xi) - T(\xi) z_1^\nu, \tag{4.46}$$

$$z_2 \geq 0, \upsilon^+ \geq 0, \upsilon^- \geq 0. \tag{4.47}$$

where $e^T = (1, 1, ..., 1)$.

If the optimal value ω' for a scenario is positive, define ϱ^ν as dual variable associated with (4.46). Set $w_1 = w_1 + 1$, generate the feasibility cut (4.41) using $W_{w_1} = (\varrho^\nu)^T T(\xi)$ and

$d_{w_1} = (\varrho^\nu)^T h(\xi)$, and add it to the master problem. If no feasibility cuts are added at this step, i.e., $\omega' \leq 0$ for every scenario ξ, then go to Step 3, otherwise go to Step 1.

Step 3. For $o = 1, ..., |\Xi|$, solve the linear program SSP and suppose π_o^ν is the optimal dual values for scenario o. If

$$\theta_o^\nu < p_o(\pi_o^\nu)^T(h_o - T_o z^\nu), \tag{4.48}$$

set $\gamma_{(l_1,o)} = 1$; otherwise set $\gamma_{(l_1,o)} = 0$.

After solving Program SSP for each scenario, define: $E_{q_1}^1 = \sum_{o=1}^{|\Xi|} p_o\gamma_{(l_1,o)}(\pi_o^\nu)^T T_o$ and $e_{q_1}^1 = \sum_{o=1}^{|\Xi|} p_o\gamma_{(l_1,o)}(\pi_o^\nu)^T h_o$; $E_{q_2}^2 = \sum_{o=1}^{|\Xi|} p_o(1 - \gamma_{(l_1,o)})(\pi_o^\nu)^T T_o$ and $e_{q_2}^2 = \sum_{o=1}^{|\Xi|} p_o(1 - \gamma_{(l_1,o)})(\pi_o^\nu)^T h_o$. If $\sum_{o=1}^{|\Xi|} \gamma_{(l_1,o)} = 0$, stop; (z^ν, θ^ν) is an optimal solution. Otherwise, set $q_1 = q_1 + 1$, set $q_2 = q_2 + 1$ if $\sum_{o=1}^{|\Xi|} \gamma_{(l_1,o)} < |\Xi|$, and add the optimality cuts to the master problem and return to Step 1.

Compared with optimality cuts in the single-cut version, our new optimality cuts (4.42) and (4.43) can be expected to be more effective since they will yield tighter bounds. The reason is that at each iteration we separate the scenarios and generate a stronger cut.

The following proposition shows that our algorithm will converge:

Proposition 19. *When Ξ is finite, the double-cut L-shaped method will converge.*

Proof. When ξ is a finite random variable, the power set of Ξ includes only finite elements. At each iteration, the double cut version cuts off at least one element of this power set. Therefore, the conclusion follows. □

4.4 FEASIBILITY CUTS TO GUARANTEE THE FEASIBILITY OF PROGRAM SSP

From the above description, we can see that at each iteration, we add a cut either to obtain a higher bound on the master problem or to restrict the feasible region of the master problem in order to make the second-stage problem (Program SSP) feasible. Because there could be many such feasibility cuts, and consequently many iterations, it could take a significant number of iterations and a long time to solve the feasibility problems to obtain these cuts before reaching optimality.

In this chapter, we try to improve the algorithm by replacing a potentially exponential number of the L-shaped feasibility cuts with a polynomially bounded number of new feasibility constraints. These new constraints will be generated *a priori* without dual information from the subproblem. We first introduce some notation. Let N be the total number of variables in Program SP and define U, the set of feasible points of SP, as follows:

$$U = \left\{(z, x, y, r, s) \in \mathbb{R}^N \middle| (4.21), (4.22), (4.23), (4.24), (4.25)\right\}. \tag{4.49}$$

Define κ_2 as the second-stage feasibility set such that $\kappa_2 = \{z|\mathcal{Q}(z) < \infty\}$. Let $\kappa_2' = \kappa_2 \cap \mathbb{B}^{|I|}$.

One interesting relationship among these sets is between U and κ_2'. Let us project the binary variables z belonging to U onto a subspace $\mathcal{Z}$ where $\mathcal{Z} = \{(z, 0)|z \in \mathbb{R}^{|I|}, (z, 0) \in \mathbb{R}^N\}$. Define $proj_{\mathcal{Z}}(U)$ to be the projection of U on $\mathcal{Z}$, i.e. $proj_{\mathcal{Z}}(U) = \{q' \mid q'$ is the projection of q on $\mathcal{Z}$ for some $q \in U\}$. Equivalently, if we define $\mathcal{Z}^{\perp} = \{(z, x(\xi), y(\xi), r(\xi), s(\xi)) \in \mathbb{R}^N|z = 0, z \in \mathbb{R}^{|I|}\}$, then for any $q = (z, x(\xi), y(\xi), r(\xi), s(\xi)) \in U$, $\exists (z', 0) \in \mathcal{Z}$ such that $q - (z', 0) \in \mathcal{Z}^{\perp}$.

Then we can obtain the following propositions:

Proposition 20. *$\kappa_2' = proj_{\mathcal{Z}}(U)$.*

Proof. 1) "$proj_{\mathcal{Z}}(U) \subseteq \kappa_2'$": Suppose $proj_{\mathcal{Z}}(U)$ is nonempty and consider $(z, 0) \in proj_{\mathcal{Z}}(U)$. By the definition of projection there exist $x(\xi), y(\xi), r(\xi), s(\xi)$ such that $q = (z, x(\xi), y(\xi), r(\xi), s(\xi)) \in U$. By the definition of U, it follows that q is feasible in U, so $\mathcal{Q}(z) < \infty$.

2) "$\kappa_2' \subseteq proj_{\mathcal{Z}}(U)$": Now suppose U is feasible. Then there exists $q = (z, x(\xi), y(\xi), r(\xi), s(\xi)) \in U$. Project q onto the subspace $\mathcal{Z}$. Then there exists $(z', 0) \in proj_{\mathcal{Z}}(U)$ such that $q - (z', 0) \in \mathcal{Z}^{\perp}$. By the definition of $\mathcal{Z}^{\perp}$, it follows that $z - z' = 0$, i.e. $z = z'$. Thus, $(z, 0) \in proj_{\mathcal{Z}}(U)$. $\square$

Following the above proposition, we introduce a method to obtain these polynomially bounded number of feasibility constraints. The advantages of these new constraints are obvious. First, we do not need to generate feasibility problems to obtain these feasibility constraints. Also, in the worst case, the total number of the constraints is far fewer. Furthermore, since they can be generated once the problem parameters are given, we can develop techniques to eliminate any dominated constraints. With these advantages, we would expect the new feasibility constraints to be much more compact and effective. We present these new constraints in the following section.

4.4.1 New Feasibility Constraints

Suppose under a scenario ξ, the set of raw stock sizes $S(\xi)$ across all facilities has $t(\xi)$ elements. That is, $S(\xi) = (r_1(\xi), r_2(\xi), ..., r_h(\xi), ..., r_{t(\xi)}(\xi))$, where $r_h(\xi)$ represents the size of the h^{th} raw under scenario ξ. Let $S'(\xi) = \{0\} \cup S(\xi)$ where $r_0 = 0$. Without loss of generality, we assume the elements in $S'(\xi)$ satisfy the following order:

$$r_0 < r_1(\xi) < r_2(\xi) < ... < r_h(\xi) < ... < r_{t(\xi)}(\xi).$$

Define $I_{r_h}(\xi) \subset I, h = 0, 1, 2, ..., t(\xi) - 1$ under scenario ξ as the set of facilities such that each facility in this set has at least one raw size longer than $r_h(\xi)$. We develop the following constraints to ensure the feasibility of the second-stage problem (Program SSP).

Proposition 21. *The vector z from the master problem can guarantee the feasibility of the second-stage problem for all scenarios $\xi(\in \Xi)$ if, and only if, it satisfies the following constraints:*

$$\sum_{i \in I_{r_h(\xi)}} M_i(\xi) z_i \geq \sum_{j \in J} \sum_{(k \in D_j(\xi), k > r_h(\xi))} \lambda_{jk}(\xi), \forall \xi \in \Xi, h = 0, 1, ..., t(\xi) - 1. \tag{4.50}$$

Proof. Section 3.3.2 has proven that, for one specific scenario ξ and its demand $\lambda(\xi)$, the vector z obtained from the master problem will guarantee the feasibility of the second-stage problem if, and only if, it satisfies constraint (4.50) corresponding to scenario ξ. Therefore, since constraint (4.50) includes all scenarios, the conclusion follows. □

Now suppose that the stock raw sizes and the capacity at each facility do not vary by scenario. Assume there are t elements in the raw set S, $S = (r_1, r_2, ..., r_t)$. Add an element $r_0 = 0$ to raw set S and define it as S'. Without loss of generality, we also assume the elements in S' satisfy the following order:

$$r_0 < r_1 < r_2 < ... < r_t.$$

Define $I_{r_h} \subset I, h = 0, 1, 2, ..., t-1$ as the set of facilities such that, each facility in this set has at least one raw size longer than r_h.

Observe that all demands with size k such that $r_h < k \leq r_{h-1}, h = 0, 1, ..., t-1$ can be satisfied by raw size $r_{h+1}, ..., r_t$. Then we can present the following proposition to ensure the feasibility of the second-stage problem.

Proposition 22. *If the capacity and raw sizes at each facility do not change under different scenarios, then vector z from the master problem can guarantee the feasibility of the second-stage optimality subproblem for all scenarios $\xi(\in \Xi)$ if, and only if, it satisfies the following constraints:*

$$\sum_{i \in I_{r_h}} M_i z_i \geq \max_{\xi \in \Xi} \sum_{j \in J} \sum_{(k \in D_j(\xi), k > r_h)} \lambda_{jk}(\xi), \forall h = 0, 1, ..., t-1. \tag{4.51}$$

Proof. Proposition 21 has shown that the vector z obtained from the master problem will guarantee the feasibility of all optimality subproblems if, and only if, it satisfies all of the following constraints:

$$\sum_{i \in I_{r_h}} M_i z_i \geq \sum_{j \in J} \sum_{(k \in D_j(\xi), k > r_h)} \lambda_{jk}(\xi), \forall h = 0, 1, ..., t-1, \xi \in \Xi. \tag{4.52}$$

It is obvious that the constraints (4.52) will be satisfied if, and only if, the vector z satisfies:

$$\sum_{i \in I_{r_h}} M_i z_i \geq \max_{\xi \in \Xi} \sum_{j \in J} \sum_{(k \in D_j(\xi), k > r_h)} \lambda_{jk}(\xi), \forall h = 0, 1, ..., t-1, \tag{4.53}$$

from which the conclusion follows. □

To illustrate the constraints in Proposition 22, consider a system with two facilities, two demand locations and two scenarios. The raw sizes available at each facility along with the unit costs are summarized in Table 15. This table also provides the unit salvage values and unit cutting cost at each facility. The demanded sizes and unit shipping costs, along with the corresponding demand rates at each demand point are summarized in Table 16 for scenario one and in Table 17 for scenario two.

Table 15: Facility Parameters

Facility 1				Facility 2			
Raw Size	Cost	Unit Scrap Value	Cutting Cost	Raw Size	Cost	Unit Scrap Value	Cutting Cost
13 7	20 12	0.05	0.05	7	13	0.1	0.06

In this example, there are two original unique raw sizes. After adding 0 to the raw set, we obtain $S' = (0, 7, 13)$. Namely, $r_0 = 0, r_1 = 7, r_2 = 13$. The corresponding facility sets feasible for $r_h (h = 0, 1)$ are as follows

$$I_{r_1} = I_7 = (1),$$

Table 16: Unit Shipping Costs under Scenario One

	Location 1		Location 2			
Size Demand	5 20	6 20	3 10	5 30	7 15	9 25
Facility 1 Facility 2	2 3	4 3	2 1.5	5 3	6 4	7 5

Table 17: Unit Shipping Costs under Scenario Two

	Location 1		Location 2	
Size	4	6	4	5
Demand	50	10	60	20
Facility 1	2	1	2	3
Facility 2	3	2	3	4

$$I_{r_0} = I_0 = (1, 2).$$

A unique demand size k with $r_1 < k \leq r_2$ is 9 for scenario one, while there is no such that size for scenario two; unique demand sizes with $r_0 < k \leq r_1$ are 3, 5, 6, 7 for scenario one and 4, 5, 6 for scenario two.

Then, since there are 2 original unique raw sizes, we have two feasibility constraints as follows:

$$\sum_{i \in I_{r_1}} M_i z_i = M_1 z_1 \geq \max_{\xi=1,2} \sum_{j \in J} \sum_{(k \in D_j(\xi), k > r_1)} \lambda_{jk}(\xi) = \max(25, 0) = 25, \tag{4.54}$$

$$\sum_{i \in I_{r_0}} M_i z_i = M_1 z_1 + M_2 z_2 \geq \max_{\xi=1,2} \sum_{j \in J} \sum_{(k \in D_j(\xi), k > r_0)} \lambda_{jk}(\xi) = \max(120, 140) = 140. \tag{4.55}$$

i.e.,

$$M_1 z_1 \geq 25, \tag{4.56}$$

$$M_1 z_1 + M_2 z_2 \geq 140. \tag{4.57}$$

Now, let us check whether these two constraints can guarantee the feasibility of the problem. First, consider constraint (4.57). This constraint ensures that Facilities 1 and 2 together can produce at least 140 units. These units can be used to satisfy any demand sizes. First consider scenario one, since the total demand across all sizes is 120 units, if the open facilities have feasible raw materials for demand size k, then the demand for this size k will always be satisfied. Whether we open Facility 1, 2, or both, there is sufficient feasible raw material to satisfy demand for sizes 3, 5, 6, 7. Based on the same reason, all demands for sizes 4, 5, 6 can also always be satisfied for scenario two since their total demand is 140 units. However, for size 9, under scenario one, constraint (4.57) alone is not sufficient to ensure that demand is met, and we also need constraint (4.56).

4.4.2 Feasibility after Using New Feasibility Constraints

Since (a) the vector z that satisfies constraints (4.51) can ensure the feasibility of the second-stage problem (Program SSP) for all scenarios, and (b) compared with constraints (4.6), these constraints are easier to generate, we can replace constraints (4.6) with constraints (4.51) in the master problem.

We know that adding new constraints into the master problem will reduce the feasible region for vector z. Therefore, before using constraints (4.51), we must ensure that they will not eliminate any points in the original feasible region for Program SP. To show this, we present the following proposition.

Proposition 23. *If z is feasible for the stochastic problem (Program SP), then it is also feasible for the feasibility constraints (4.51).*

Proof. Consider any arbitrary h, and without loss of generality, suppose the maximum right hand side of constraints 4.51 is from scenario ξ'. From constraint (4.22), it follows that

$$\sum_{j\in J}\sum_{(k\in D_j(\xi'),k>r_h)}\lambda_{jk}(\xi')=\sum_{j\in J}\sum_{(k\in D_j(\xi'),k>r_h)}\sum_{i\in I}x_{ijk}(\xi')=\sum_{i\in I}\sum_{j\in J}\sum_{(k\in D_j(\xi'),k>r_h)}x_{ijk}(\xi'). \tag{4.58}$$

Recalling the definition of I_{r_h}, we can rewrite the above equation (4.58) as

$$\sum_{j\in J}\sum_{(k\in D_j(\xi'),k>r_h)}\lambda_{jk}(\xi')=\sum_{i\in I}\sum_{j\in J}\sum_{(k\in D_j(\xi'),k>r_h)}x_{ijk}(\xi')=\sum_{i\in I_{r_h}}\sum_{j\in J}\sum_{(k\in D_j(\xi'),k>r_h)}x_{ijk}(\xi'). \tag{4.59}$$

Because $\sum_{(k\in D_j(\xi'),k>r_h)}x_{ijk}(\xi')\leq\sum_{k\in D_j}x_{ijk}(\xi')$, from equation (4.59), it follows that

$$\sum_{j\in J}\sum_{(k\in D_j(\xi'),k>r_h)}\lambda_{jk}(\xi')\leq\sum_{i\in I_{r_h}}\sum_{j\in J}\sum_{k\in D_j}x_{ijk}(\xi'). \tag{4.60}$$

From (4.24) and (4.60), we conclude that

$$\sum_{j\in J}\sum_{(k\in D_j(\xi'),k>r_h)}\lambda_{jk}(\xi')\leq\sum_{i\in I_{r_h}}\sum_{j\in J}\sum_{k\in D_j}x_{ijk}(\xi')\leq\sum_{i\in I_{r_h}}M_iz_i. \tag{4.61}$$

Since ξ' yields the maximum value for $\sum_{j\in J}\sum_{(k\in D_j(\xi),k>r_h)}\lambda_{jk}(\xi)$, the conclusion follows.

□

Although constraints (4.51) may be fewer in number than the L-shaped feasibility cuts, there are still some dominated constraints. There are three kinds of dominated constraints among them. We can use the technique described in Chapter 3 to identify them and eliminate the redundancy. After these eliminations, we note that the number of our feasibility constraints (4.51) is polynomial since it consists of no more than t (the number of unique raw sizes) constraints. Now, we can replace constraints (4.6) by non-dominated constraints (4.51).

Double Cut Master Problem with *a priori* Feasibility Cuts (Program DMP)

$$\min\sum_{i\in I}f_iz_i+\sum_{o=1}^{|\Xi|}\theta_o \tag{4.62}$$

subject to:

$$\sum_{i\in I_{r_h}}M_iz_i\geq\max_{\xi\in\Xi}\sum_{j\in J}\sum_{(k\in D_j(\xi),k>r_h)}\lambda_{jk}(\xi),\forall h=0,1,...,t-1, \tag{4.63}$$

$$E_{l_1}z+\sum_{o=1}^{|\Xi|}\gamma_{(l_1,o)}\theta_o\geq e_{l_1},l_1=1,...,q_1, \tag{4.64}$$

$$E_{l_2}z+\sum_{o=1}^{|\Xi|}(1-\gamma_{(l_1,o)})\theta_o\geq e_{l_2},l_2=1,...,q_2, \tag{4.65}$$

$$\theta\in\Re,z_i\in\{0,1\},i\in I. \tag{4.66}$$

4.5 ALTERNATIVE L-SHAPED METHODS

In the above section, we presented a new version of the feasibility constraints. In practice, after eliminating the dominated constraints, our feasibility constraints (4.63) are much more effective than the dual-based feasibility cuts (4.63) because (a) there are fewer constraints, (b) there is no redundancy, (c) more than one constraint is added at each iteration, and (d) these are available without generating and solving the feasibility problem SFP.

One easy way to make use of the feasibility constraints in the above section is to replace the double cut master problem with Program DMP, which includes all feasibility constraints (4.63). We can present this new version of the L-shaped method (Algorithm 5) as follows.

Algorithm 5 Double Cut L-shaped Method with *a priori* Feasibility Cuts

```
1: Set $q_1 = q_2 = \nu = 0$.
2: Generate the feasibility constraints (4.63) and add them into the master problem to
   obtain the initial new master problem (Program DMP).
3: Set $\nu = \nu + 1$ and solve problem DMP.
4: Solve LP problem SSP for scenario $o = 1, ..., |\Xi|$ described in Section 4.1.1. Let $\pi_o^\nu$ be
   the associated optimal dual values for scenario $o$ and if
       $$\theta_o^\nu < p_o(\pi_o^\nu)^T(h_o - T_o z^\nu), \quad (4.67)$$
   set $\gamma_{(l_1,o)} = 1$; otherwise set $\gamma_{(l_1,o)} = 0$.
   After solving all SSP for all scenarios, define: $E_{q_1}^1 = \sum_{o=1}^{|\Xi|} p_o \gamma_{(l_1,o)} (\pi_o^\nu)^T T_o$ and
   $e_{q_1}^1 = \sum_{o=1}^{|\Xi|} p_o \gamma_{(l_1,o)} (\pi_o^\nu)^T h_o$; $E_{q_2}^2 = \sum_{o=1}^{|\Xi|} p_o (1 - \gamma_{(l_1,o)})(\pi_o^\nu)^T T_o$ and $e_{q_2}^2 = \sum_{o=1}^{|\Xi|} p_o (1 -
   \gamma_{(l_1,o)})(\pi_o^\nu)^T h_o$. If $\sum_{o=1}^{|\Xi|} \gamma_{(l_1,o)} = 0$, stop; $(z^\nu, \theta^\nu)$ is an optimal solution. Otherwise, set
   $q_1 = q_1 + 1$, set $q_2 = q_2 + 1$ if $\sum_{o=1}^{|\Xi|} \gamma_{(l_1,o)} < |\Xi|$, add the optimality cuts to the master
   problem, and then go to step 3.
```

From the above, we can see that for Algorithm 5 all feasibility constraints are generated at the beginning. Adding all of them into the master problem DMP will increase its objective function value and obtain a better lower bound. However, it is very possible that we do not need all feasibility constraints to achieve optimality since our objective of adding feasibility constraint is only to help find the optimal solution, not all feasible points. Therefore, in

Algorithm 5 there is a tradeoff between the time to add new feasibility constraints and the time to solve the larger master problem. There is also a tradeoff between fewer iterations being required but a longer time being required to solve the master problem at each iteration.

One idea is only to select and add the violated feasibility constraints. In order to do this, we should choose and add the feasibility constraints based on the information provided by the master problem at each iteration. That is, at each iteration, we first solve the master problem to obtain the optimal z values. Then, we separate the feasibility constraints by checking whether they are violated by the optimal z values, and only add the violated constraints into the master problem. Based on this idea, we present an alternative new algorithm (Algorithm 6) as follows:

Algorithm 6 Double Cut L-shaped Method with Feasibility Cuts Added as Needed

```
1: Set $q_1 = q_2 = \nu = 0$ and set the master problem as the initial master problem DMP.
2: Generate all feasibility constraints (4.63). Suppose there are $b$ total constraints and set $B = \{1, 2, ..., b\}$. Let ${}^*z = (1, ..., 1)^{|I|}$ be the optimal solution to the problem DMP and go to step 5.
3: Set $\nu = \nu + 1$ and solve the problem DMP.
4: Check each feasibility constraint $t, t \in B$. If it is violated, add it to the problem DMP and set $B = B \backslash t$; if it is satisfied, ignore it. If any constraint is added to the problem DMP, go to step 3, otherwise go to step 5.
5: Solve LP problem SSP for scenario $o = 1, ..., |\Xi|$ described in Section 4.1.1. Let $\pi_o^\nu$ be the associated optimal dual values for scenario $o$ and if
       $$\theta_o^\nu < p_o(\pi_o^\nu)^T(h_o - T_o z^\nu), \tag{4.68}$$
   set $\gamma_{(l_1,o)} = 1$; otherwise set $\gamma_{(l_1,o)} = 0$.
   Then define: $E_{q_1}^1 = \sum_{o=1}^{|\Xi|} p_o \gamma_{(l_1,o)} (\pi_o^\nu)^T T_o$ and $e_{q_1}^1 = \sum_{o=1}^{|\Xi|} p_o \gamma_{(l_1,o)} (\pi_o^\nu)^T h_o$; $E_{q_2}^2 = \sum_{o=1}^{|\Xi|} p_o (1 - \gamma_{(l_1,o)}) (\pi_o^\nu)^T T_o$ and $e_{q_2}^2 = \sum_{o=1}^{|\Xi|} p_o (1 - \gamma_{(l_1,o)}) (\pi_o^\nu)^T h_o$. If $\sum_{o=1}^{|\Xi|} \gamma_{(l_1,o)} = 0$, stop; $(z^\nu, \theta^\nu)$ is an optimal solution. Otherwise, set $q_1 = q_1 + 1$, set $q_2 = q_2 + 1$ if $\sum_{o=1}^{|\Xi|} \gamma_{(l_1,o)} < |\Xi|$, and add the optimality cuts to the master problem and return to step 3.
```

To check how these new feasibility cuts affect the performance of the traditional single-cut L-shaped method, we also develop two algorithms by replacing the L-shaped feasibility cuts with non-dominated feasibility constraints (4.63). We first present the algorithm in which we add all feasibility constraints (4.63) *a priori* (Algorithm 7).

Algorithm 7 Single-Cut L-shaped Method with *a priori* Feasibility Cuts

```
1: As before in the single-cut L-shaped method.
2: Generate all feasibility constraints (4.63) and add them into the master problem to obtain the initial new master problem.
3: Set $\nu = \nu + 1$ and solve the resulting master problem.
4: Solve optimality subproblem for $\xi = 1, ..., |\Xi|$ as before in the single-cut L-shaped method. Stop if optimal. Otherwise add the optimality cut (4.7) to master problem and return to Step 3.
```

Similar to Algorithm 6, we can also add the feasibility cuts only when needed for the single-cut L-shaped method. We present this algorithm as follows (Algorithm 8):

Algorithm 8 Single-Cut L-shaped Method with Feasibility Cuts Added as Needed

```
1: As before in the single-cut L-shaped method.
2: Generate all feasibility constraints (4.63). Suppose there are $b$ total constraints and set $B = \{1, 2, ..., b\}$. Let ${}^*z = (1, ..., 1)^{|I|}$ be the optimal solution to the master problem and go to step 5.
3: Set $\nu = \nu + 1$ and solve the resulting master problem.
4: Check each feasibility constraint $t, t \in B$. If it is violated, add it to the master problem and set $B = B \backslash t$; if it is satisfied, ignore it. If any constraint is added to the master problem, go to step 3, otherwise go to step 5.
5: Solve optimality subproblem for $\xi = 1, ..., |\Xi|$ as before in the single-cut L-shaped method. Stop if optimal. Otherwise add the optimality cut (4.7) to master problem and return to Step 3.
```

4.6 COMPUTATIONAL RESULTS

In this section, we compare the effectiveness of the above algorithms by doing some computational tests. We first compare the performances of the following six algorithms:

- Single-cut L-shaped method: add single optimality cut and single traditional L-shaped feasibility cut per iteration.
- Double-cut L-shaped method: add double optimality cuts and single traditional L-shaped feasibility cut per iteration.
- Algorithm 5: add double optimality cuts per iteration and all feasibility constraints (4.63) *a priori*.
- Algorithm 6: add double optimality cuts and feasibility constraints (4.63) when needed per iteration.
- Algorithm 7: add single optimality cut per iteration and all feasibility constraints (4.63) *a priori*.
- Algorithm 8: add single optimality cut and feasibility constraints (4.63) when needed per iteration.

Their performances are measured by solving three sets of instances: 1) small problems, 2) medium problems, and 3)large problems. There are 20 instances in each set. These instances are randomly generated with parameters whose ranges are showed in Table 18.

Table 18: Characteristics of Tested Stochastic Instances

Instance Set	*No. of Facilities*	*No. of Customers*	*No. of Raw Sizes per Facility*	*No. of Demand Sizes per Customer*	*No. of Scenarios*
Small	2 to 4	3 to 8	1 to 4	2 to 7	20 to 80
Medium	5 to 15	9 to 30	2 to 7	3 to 12	81 to 180
Large	16 to 40	31 to 80	3 to 10	4 to 18	181 to 350

We use each of the six algorithms to solve 20 problems (Program SP) in each set. To evaluate an algorithm's performance, we look at the number of instances where the algorithm

Table 19: Performance Comparison for Stochastic Algorithms

	Single-Cut Method				*Double-Cut Method*			
Instance Set	*L-shaped Fea*	*All Fea a priori*	*Fea When Needed*	*Total*	*L-shaped Fea*	*All Fea a priori*	*Fea When Needed*	*Total*
Small	0	8	4	12	0	5	3	8
Medium	0	1	4	5	0	3	12	15
Large	0	0	3	3	0	1	16	17

solves the problem in the least amount of time. To emphasize the comparison between the single-cut and the double-cut method, we add the item "Total" in the following table. Total represents the total number of the problems where the single-cut / double-cut method wins.

We summarize the computational results in Table 19; a complete list of the results can be found at Appendix C. From Table 19 we can make the following observations:

(1) When comparing the procedures for generating the optimality cuts, the performances by the single-cut and the double-cut methods seems to depend on the sizes of the problems. For the set of small problems, the single-cut method performs better than double-cut method: among 20 instances, the single-cut method wins 12. The reason for this is that, for small problems, the number of iterations for both methods is small and the numbers are roughly the same. Since the double-cut method adds two cuts at each iteration, the master problem is larger and it usually takes longer to solve the problem.

When the size of the problem increases, the double-cut method performs better than the single-cut method: for the instances in the set of medium problems, the double-cut method wins 15 out of total 20 instances; for large-size problems, it beats the single-cut method by winning 17 instances. This is mainly because when the problem becomes large, the double-cut method will take significantly fewer iterations than the single-cut method, which reduces the solution time because there are fewer iterations (at each iteration, each algorithm has to solve at least $|\Xi|$ subproblems)and furthermore the master problem is smaller as well. We may thus conclude that the double-cut method will perform better than single-cut method when the problems become bigger.

(2) For the approach used to add the feasibility constraints / cuts, the different methods in the single-cut method and the double-cut method perform slightly differently, but the trends reflected by these two methods are similar.

In general the "*a priori*" approach is the better for small problems, while the "as-needed" approach is better for large problems. Looking across the single and double optimality cut methods, the "*a priori*" approach (Algorithms 5 and 7) wins 13 out of 20 for the small problems, but the "as-needed" approach (Algorithms 6 and 8) wins 16 out of 20 and 19 out of 20 for the medium and large problems, respectively.

Small problem will, in general, not have many feasibility cuts and adding all of them into the master problem will not make much difference. The time savings from the fewer iterations required to solve the problem and not having to check which feasibility cuts should be added outweigh the extra time required for the larger problems.

However, when the problems become larger, there will be more scenarios, along with more different kinds of raw materials and demand. Therefore, we need more feasibility cuts to guarantee the feasibility of all subproblems. We definitely do not want to add unnecessary feasibility cuts into the master problem, since it will make the master problem much larger and harder to solve at each iteration. This is why the method of adding feasibility cuts when needed becomes more and more effective with increasing size of the problem.

(3) The traditional L-shaped feasibility cuts perform the worst and the algorithms based on this method never win for any instance. This implies that our new feasibility constraints (4.63) are much better than the traditional L-shaped feasibility cuts. The main reason is that, compared with the non-dominated feasibility constraints in (4.63), many of the traditional L-shaped feasibility cuts generated are dominated(some feasibility cuts generated earlier are dominated by cuts generated later on). In other words, the traditional L-shaped method generates far more than the necessary number of feasibility cuts and adds all of them into the master problem. A second reason is that it only can generate one cut at each iteration and thus takes many iterations to obtain these cuts. A third reason is that it needs to solve

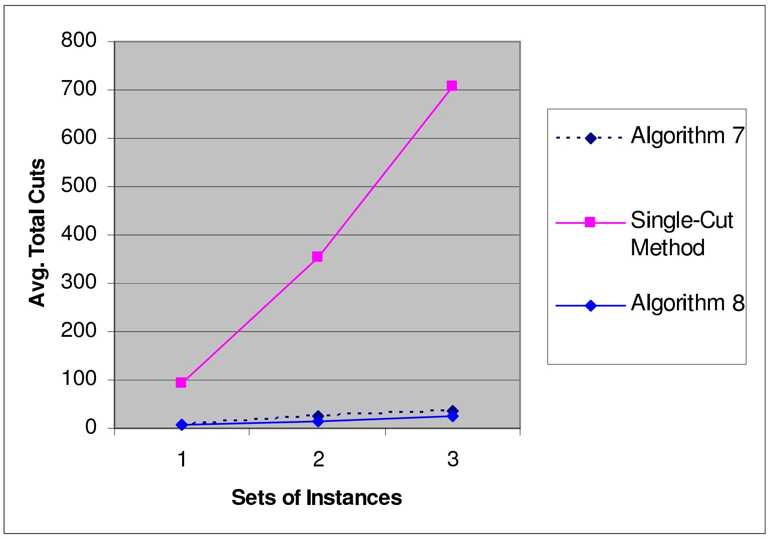

Figure 8: Comparison of Total Cuts

$o(= |\Xi|)$ feasibility subproblems at each iteration to obtain the feasibility cut. In contrast, our feasibility constraints (4.63) do not need to be obtained by solving any feasibility problem and are all known beforehand.

(4) Finally, as the problems become larger, all algorithms need more cuts and iterations to obtain the optimal solutions. However, the rate of increase for algorithms with our feasibility constraints (4.63) is significantly smaller. Figure 8 illustrates this point further. In the figure, single-cut method represents the traditional single-cut L-shaped method.

In conclusion, Algorithm 6 (double-cut and feasibility constraints (4.63) as needed per iteration) appears to be the best among these six algorithms, especially for the large problems. Not only does it eliminate dominated constraints, but it also needs less time, fewer iterations and a fewer total number of feasibility and optimality cuts to solve the NP-hard stochastic problem. The results in Table 19 clearly support this conclusion: compared with other methods, the number of large instance won by Algorithm 6 is significantly larger.

5.0 CONCLUSIONS AND FUTURE RESEARCH

5.1 CONCLUSION

This research has addressed a stochastic location-production-distribution problem for remnant inventory supply chains through three models. The development builds upon and generalizes a model first introduced by Adelman and Nemhauser [2, 3], who addressed a related problem with a single supply and a single demand location in the context of the fiber-optic cable industry. Our problem is motivated by an application in the steel industry where the problem environment is significantly more complex. We consider positive cutting costs, nonzero scrap values, the presence of multiple facilities and multiple sources of demand, fixed facility costs, capacity limits on facilities and the selection of facilities to operate.

The first model addresses only production and distribution. A linear programming model is developed, and several new theoretical results are provided to develop an operating policy that can be used in a stochastic and dynamic environment such as the one underlying the problem that motivated this research. Results from a simulation study indicate that this policy performs significantly better than other plausible heuristic procedures.

In the second model, we integrate the capacitated facility location problem into the previous model. We show that the general policy developed in the previous model to manage the cost under a dynamic environment still applies to this new situation. We also develop an alternative method to generate feasibility constraints that are non-dominated and, in the worst case, polynomially bounded in number. The feasibility cuts in Benders' decomposition are replaced by these constraints. We prove that these constraints ensure the feasibility of the second-stage problem, while not eliminating any feasible points. We also conduct a comparative polyhedral study to show that the new feasibility constraints have the same convex hull as that of Benders' feasibility cuts. Finally, computational results show that our new algorithm based on the feasibility constraints is much more efficient than the traditional Benders' decomposition algorithm.

In the third model, we consider the most general case: a stochastic problem combining facility location, production, and distribution. We use the L-shaped method to solve this problem. In order to improve this method, we present a new way to generate optimality cuts and a new method to obtain all feasibility constraints without solving the feasibility problem. We find that these optimality cuts generate better bounds for the master problem. We also conclude that these new feasibility constraints are much more effective, by showing that compared with the traditional L-shaped feasibility cuts, the total number of these new feasibility constraints is polynomially bounded. Also, there are no dominated feasibility constraints, and while more feasibility constraints might be added into the master problem per iteration, the total number of these required to reach optimality is smaller. Based on these new optimality and feasibility constraints, we develop the double-cut L-shaped method. Computational tests show that our double-cut L-shaped method with feasibility cuts when needed takes fewer iterations, adds fewer cuts in total, and obtains optimal solutions in less time. Furthermore, from the computational results in Chapters 3 and 4, we conclude that our new feasibility constraint is robust for the different methodologies. It is especially effective for the stochastic problem and it performs better when the problem becomes larger.

5.2 EXTENSIONS AND FUTURE WORK

In future research, we could first extend our master models to consider the two-dimensional cutting problem. Such problems arise in industries such as glass, copper, paper, etc. For this model, we would have to find an intelligent way to measure the inherent value for any remnant area in order to develop a price-directed policy. In the two-dimensional problem, the inherent value for a remnant may not simply be the dual value for a single variable. We might need to combine some dual values to obtain the inherent value for a remnant area. Therefore, it is likely to be a more complicated extension.

Second, in this research, we do not integrate inventory control directly into the models. Instead, we considered a heuristic approach to this problem. This could be a shortcoming since the inherent value policy in the master models does not consider the inventory holding costs. In practice, for some industries, the inventory holding costs may not be trivial so that this cannot be ignored; future research should aim at integrating the inventory control problem into the model. Finding a suitable price-directed policy for this more complicated model will be a challenge. This is because in the new model, the inherent values for the different sizes should somehow reflect the change in the different inventory levels in stock. For example, when the inventory for one remnant is going up, its inherent value should be going down. Also, we assume in this research that the raw suppliers are integrated into the distribution centers. This may not be the case, and future research could separate the raw suppliers from the distribution centers.

Third, in the stochastic programming model in this research, we have separated the optimal scenarios from those that are not optimal. However, we only divide them into two groups to generate two optimality cuts. We believe a cutting-plane technique could be applied to build more effective optimality cuts. Also, among all scenarios, there may be more than two different characteristics. In other words, we can divide them into more different groups to build more powerful optimality cuts. For example, we can study the properties of the polyhedra for optimality problem SSP, then we can aggregate the scenarios with similar properties into a group to generate a cut. However, this needs to be studied further. Also, the double cut method should be tested further on general stochastic programming problems to see how it performs relating to the single and multi-cut.

Finally, because the mixed-integer programs in this research are NP-hard, it could be worth developing heuristics for the location model and the stochastic problem to decide which facilities to open, and to obtain inherent value for remnants at each facility. This is because for NP-hard problems, it is still quite time-consuming for methods like Benders' decomposition and the L-shaped method to solve the problem. It is also a nontrivial task to formulate and solve the perturbation to obtain the non-degenerate inherent value. In addition to developing these heuristics, we perform a computational study to examine whether the inherent values obtained from the heuristics are close to the optimal dual prices.

APPENDIX A

COMPUTATIONAL RESULTS FOR LP MODEL

Table 20: Performance of Three Policies and Bound Gap between Proposition 11 and Optimal Values

Set	*Instance*	*Our Policy*	*Smallest Fit*			*Multi-criteria*			*Bound Gap*
			Max	*Min*	*Avg*	*Max*	*Min*	*Avg*	
Small	1	1.00	1.26	1.25	1.26	1.18	1.18	1.18	6.46%
	2	1.00	1.27	1.27	1.27	1.09	1.09	1.09	5.78%
	3	1.00	1.45	1.44	1.44	1.17	1.16	1.17	3.29%
	4	1.00	1.18	1.18	1.18	1.04	1.04	1.04	2.15%
	5	1.00	1.33	1.32	1.33	1.20	1.19	1.19	10.76%
	6	1.00	1.53	1.52	1.53	1.15	1.15	1.15	2.88%
	7	1.00	1.55	1.55	1.55	1.20	1.19	1.20	12.04%
	8	1.00	1.52	1.51	1.51	1.30	1.29	1.30	7.70%
	9	1.00	1.48	1.48	1.48	1.19	1.18	1.19	9.26%
	10	1.00	1.33	1.33	1.33	1.14	1.13	1.13	10.67%
	Avg	1.00	1.39	1.39	1.39	1.17	1.16	1.16	7.10%

Table 21: Performance of Three Policies and Bound Gap, Cont.

Set	*Instance*	*Our Policy*	*Smallest Fit*			*Multi-criteria*			*Bound Gap*
			Max	*Min*	*Avg*	*Max*	*Min*	*Avg*	
Medium	1	1.00	1.43	1.42	1.43	1.16	1.16	1.16	13.00%
	2	1.00	1.91	1.21	1.78	1.22	1.22	1.22	8.00%
	3	1.00	1.68	1.24	1.54	1.58	1.57	1.57	4.99%
	4	1.00	1.73	1.18	1.41	1.64	1.64	1.64	6.89%
	5	1.00	1.74	1.31	1.49	1.57	1.57	1.57	5.89%
	6	1.00	1.51	1.24	1.40	1.38	1.38	1.38	10.70%
	7	1.00	1.88	1.39	1.67	1.20	1.19	1.20	7.60%
	8	1.00	1.31	1.31	1.31	1.17	1.17	1.17	8.64%
	9	1.00	1.68	1.30	1.52	1.41	1.41	1.41	9.96%
	10	1.00	1.48	0.94	1.13	1.38	1.38	1.38	9.22%
	Avg	1.00	1.64	1.25	1.47	1.37	1.37	1.37	8.49%
Large	1	1.00	1.98	1.98	1.98	1.74	1.74	1.74	11.65%
	2	1.00	1.98	1.31	1.69	1.37	1.36	1.37	12.03%
	3	1.00	1.84	1.84	1.84	1.58	1.35	1.50	10.46%
	4	1.00	1.68	1.40	1.54	1.73	1.67	1.69	10.86%
	5	1.00	1.82	1.56	1.73	1.54	1.54	1.54	8.84%
	6	1.00	1.96	1.33	1.70	1.39	1.39	1.39	12.92%
	7	1.00	1.70	1.65	1.67	2.20	2.13	2.17	14.93%
	8	1.00	1.59	1.35	1.51	1.85	1.85	1.85	10.15%
	9	1.00	1.61	1.39	1.54	1.67	1.67	1.67	8.38%
	10	1.00	1.95	1.95	1.95	1.64	1.64	1.64	12.94%
	Avg	1.00	1.81	1.58	1.72	1.67	1.63	1.66	11.32%

APPENDIX B

COMPUTATIONAL RESULTS FOR LOCATION MODEL

The tables in Appendix B use

- IT to denote the total number of iterations required by an algorithm to add all feasibility cuts / constraints before obtaining the optimality.
- TIm to denote the total time for an algorithm to solve the MIP problem.

All other characteristics are as defined in Chapter 3.

For the algorithm that adds all feasibility constraints *a priori*, AC and IT are not applicable and DC is always 0; for the traditional Benders' decomposition, AC is always 1 and IT is always equivalent to TC; for the algorithm that adds feasibility constraints only when needed, DC is always 0. For these cases, we do not list the results in order to save space.

Table 22: Performance on Small Location Problems

Inst-	*All a Priori*		*Benders*			*Cuts When Needed*			
ance	TC	TIM	TC	DC	TIM	TC	AC	IT	TIM
1	1	0.15	1	0	0.23	1	1	1	0.15
2	1	0.20	1	0	0.40	1	1	1	0.19
3	1	0.44	1	0	0.75	1	1	1	0.42
4	1	0.66	1	0	1.03	1	1	1	0.62
5	1	0.03	1	0	0.08	1	1	1	0.03
6	1	0.07	1	0	0.14	1	1	1	0.07
7	1	0.04	1	0	0.08	1	1	1	0.04
8	1	0.03	1	0	0.05	1	1	1	0.03
9	1	1.73	1	0	2.32	1	1	1	1.55
10	1	0.04	1	0	0.06	1	1	1	0.04
11	1	0.23	1	0	0.46	1	1	1	0.22
12	1	2.07	1	0	2.58	1	1	1	1.77
13	4	0.34	4	0	0.60	4	2	2	0.33
14	5	4.00	6	0	5.62	2	2	1	3.95
15	3	1.34	4	1	2.03	3	1	3	1.20
16	4	2.81	4	0	3.78	2	2	1	2.69
17	2	1.89	1	1	2.28	1	1	1	1.61
18	5	4.93	6	1	6.50	1	1	1	4.72
19	2	1.34	2	0	2.00	2	1	2	1.22
20	4	0.84	4	1	1.47	2	2	1	0.82

Table 23: Performance on Medium Location Problems

Inst-	*All a Priori*		*Benders*			*Cuts When Needed*			
ance	TC	TIM	TC	DC	TIM	TC	AC	IT	TIM
1	5	9.09	5	2	11.21	2	2	1	8.70
2	6	203.98	1	0	241.94	2	2	1	201.71
3	8	239.40	6	1	265.75	5	2.5	2	228.69
4	6	20.67	3	1	24.78	3	3	1	19.39
5	7	602.20	1	0	651.07	2	2	1	572.70
6	7	44.09	5	1	51.41	4	2	2	43.14
7	12	117.77	16	10	174.61	3	1.5	2	58.89
8	10	163.14	16	11	189.42	4	4	1	130.52
9	7	106.94	2	0	118.95	2	2	1	105.97
10	8	220.10	5	1	237.64	4	2	2	216.94
11	10	93.20	13	8	128.07	2	2	1	91.81
12	8	110.36	9	4	152.26	2	2	1	106.80
13	9	30.58	11	6	50.39	2	2	1	32.49
14	6	30.62	6	2	41.30	2	2	1	30.37
15	4	13.81	4	0	17.51	4	2	2	13.67
16	7	81.73	1	0	88.94	1	1	1	79.35
17	4	8.15	19	10.24	10	3	1.5	2	8.00
18	10	46.33	15	9	78.34	3	1.5	2	46.51
19	9	396.62	1	0	427.80	3	3	1	386.70
20	5	36.11	4	2	41.84	2	2	1	35.34

Table 24: Performance on Large Location Problems

Inst-	*All a Priori*		*Benders*			*Cuts When Needed*			
ance	TC	TIM	TC	DC	TIM	TC	AC	IT	TIM
1	21	4473	9	4	4999	5	2.5	2	4418
2	16	2956	1	0	4746	4	4	1	2794
3	14	523	15	11	640	3	1.5	2	337
4	12	367	15	11	411	3	1.5	2	333
5	12	331	8	4	362	3	1.5	2	259
6	17	2230	1	0	1668	1	1	1	1240
7	10	4838	1	0	11040	1	1	1	4441
8	12	349	27	19	424	4	1.33	3	367
9	8	881	1	0	926	2	2	1	843
10	13	8212	1	0	9775	2	2	1	7632
11	13	1463	2	0	1545	2	1	2	1411
12	11	3046	1	0	4748	2	2	1	2936
13	18	4599	10	4	4572	5	2.5	2	4454
14	13	3966	5	2	4306	2	2	1	3870
15	13	3944	5	2	4155	4	1.33	3	3859
16	12	3546	18	12	4101	4	1.33	3	3469
17	14	2134	5	1	2341	4	1.33	3	2083
18	17	8179	1	0	9111	2	2	1	7130
19	10	3721	1	0	4527	2	2	1	3723
20	10	493	6	3	637	2	2	1	514

APPENDIX C

COMPUTATIONS FOR STOCHASTIC MODEL

Note that in the tables of this appendix, we test the following six algorithms:

- Single-cut L-shaped method: add single optimality cut and single traditional L-shaped feasibility cut per iteration.
- Double-cut L-shaped method: add double optimality cuts and single traditional L-shaped feasibility cut per iteration.
- Algorithm 5: add double optimality cuts per iteration and all feasibility constraints (4.63) *a priori*.
- Algorithm 6: add double optimality cuts and feasibility constraints (4.63) when needed per iteration.
- Algorithm 7: add single optimality cut per iteration and all feasibility constraints (4.63) *a priori*.
- Algorithm 8: add single optimality cut and feasibility constraints (4.63) when needed per iteration.

Table 25: Performance of Algorithms for Small Stochastic Problems

Instance	*Single-Cut Method*			*Double-Cut Method*		
Set	*L-shaped Fea*	*All Fea a priori*	*Fea When Needed*	*L-shaped Fea*	*All Fea a priori*	*Fea When Needed*
1	12.26	2.11	2.44	10.46	2.63	3.00
2	14.39	2.63	3.33	11.07	2.99	3.10
3	16.05	3.52	3.19	12.11	3.43	3.20
4	19.65	4.84	4.60	17.99	4.16	4.29
5	28.35	4.61	4.56	22.40	6.11	5.83
6	13.20	4.04	3.88	13.89	2.43	2.38
7	16.03	5.16	4.60	15.87	3.06	3.01
8	8.92	3.55	3.21	7.80	6.43	6.15
9	22.50	3.54	3.38	15.63	4.71	3.85
10	17.07	5.91	6.36	11.78	4.45	4.84
11	11.68	3.19	3.75	11.55	2.08	2.17
12	11.29	3.01	3.85	9.46	3.38	3.55
13	12.53	3.01	4.21	10.64	3.39	3.51
14	6.71	1.32	1.31	6.02	1.46	1.40
15	4.20	1.91	2.53	3.93	2.37	2.42
16	15.20	3.34	4.31	12.76	3.64	4.49
17	9.31	1.23	3.21	7.65	1.32	2.52
18	14.97	5.87	6.76	11.43	4.72	5.30
19	13.87	4.22	6.23	10.54	4.89	4.50
20	36.84	5.21	4.95	31.56	3.31	3.29

Table 26: Performance of Algorithms for Medium Stochastic Problems

Instance	*Single-Cut Method*			*Double-Cut Method*		
Set	*L-shaped Fea*	*All Fea a priori*	*Fea When Needed*	*L-shaped Fea*	*All Fea a priori*	*Fea When Needed*
1	369.15	186.54	162.54	168.13	135.32	102.45
2	153.98	22.53	22.99	177.15	34.18	25.69
3	237.43	43.40	40.52	251.30	50.71	46.41
4	2737.00	611.21	561.54	1099.37	293.86	289.02
5	807.11	312.23	287.46	259.37	128.75	118.16
6	696.27	276.54	288.76	453.69	184.24	186.75
7	798.43	322.83	286.53	543.13	215.87	208.21
8	673.65	217.15	204.93	349.10	163.35	134.35
9	554.19	150.21	143.83	434.87	240.68	204.22
10	1124.95	172.96	158.71	1393.07	214.90	200.59
11	478.46	141.98	133.17	315.04	107.90	98.91
12	1105.32	368.32	354.93	732.76	211.78	183.08
13	Unsolved	423.96	434.97	Unsolved	360.16	364.24
14	356.70	142.69	135.63	365.32	151.59	137.00
15	245.72	122.49	111.63	201.63	86.73	71.65
16	547.04	226.52	232.84	385.43	180.40	184.33
17	424.32	152.75	131.31	279.88	108.20	96.91
18	1370.11	498.67	432.33	742.92	263.50	240.13
19	540.34	230.53	129.55	428.42	174.63	78.93
20	1605.76	602.11	432.35	1258.82	394.65	277.05

Table 27: Performance of Algorithms for Large Stochastic Problems

Instance	*Single-Cut Method*			*Double-Cut Method*		
Set	*L-shaped Fea*	*All Fea a priori*	*Fea When Needed*	*L-shaped Fea*	*All Fea a priori*	*Fea When Needed*
1	6734.97	1411.31	1123.55	4365.83	726.91	577.48
2	960.69	400.11	392.39	1237.43	414.10	423.68
3	500.64	184.70	168.36	323.70	119.89	106.64
4	1581.51	464.50	450.35	1286.74	478.83	408.64
5	5022.35	1022.45	767.73	3117.63	674.82	568.90
6	1273.04	559.55	556.06	943.34	430.39	406.30
7	718.86	178.95	171.95	823.54	228.27	214.82
8	1554.48	278.54	283.60	1134.64	208.12	213.24
9	2120.41	317.23	321.50	1506.92	268.27	183.28
10	6460.66	921.56	790.43	2811.64	547.04	404.62
11	754.68	197.71	113.95	956.14	202.25	158.28
12	Unsolved	458.72	380.34	Unsolved	376.56	297.65
13	3575.85	786.07	726.93	2690.26	484.16	417.58
14	1602.82	359.17	343.60	729.57	239.22	173.64
15	1852.18	367.89	361.18	1562.43	283.00	209.20
16	2142.29	432.32	412.43	1838.48	376.57	327.74
17	2276.13	845.69	764.85	1698.62	515.97	431.72
18	4546.39	720.77	559.17	2974.32	403.45	286.52
19	Unsolved	958.50	881.11	Unsolved	603.79	543.12
20	1058.69	359.42	313.73	874.38	202.89	188.28

APPENDIX D

NOMENCLATURE

D.1 NOTATION FOR DETERMINISTIC PROBLEM

D.1.1 Greek Letters

δ_i — Cost per cut at facility $i \in I$; $\delta_i > 0$.

λ_{jk} — Mean demand rate at location $j \in J$ for product of size $k \in D_j$.

σ_i — Unit salvage value of product at facility $i \in I$; $\sigma_i > 0$.

D.1.2 Roman Letters

a_{ih} — Cost of one unit of raw stock of size $h \in S_i$ at location $i \in I$.

c_{ijk} — Cost of transporting a single unit of size $k \in D_j$ from location $i \in I$ to location $j \in J$.

D_j — Set of all sizes that are demanded at demand location $j \in J$; define $D = \bigcup_{j \in J} D_j$.

f_i — Equivalent uniform daily fixed cost of operating a facility at location $i \in I$.

i — Facility location.

I — Set of facility locations.

j — Demand location.

J — Set of demand locations.

K_i — Set of all possible sizes, including scrap sizes (we refer to sizes that are smaller than the smallest demanded size as scrap sizes), that could be generated at facility i (we include the size 0 in this set); define $K = \bigcup_{i \in I} K_i$.

M_i — Total daily processing capacity for facility $i \in I$.

r_{ih} — Rate of replenishment of raw units of size $h \in S_i$ at facility location $i \in I$; define $r_{ih} = 0$ if $h \in (K_i \backslash S_i)$.

s_{iu} — Rate at which size u is scrapped at facility location $i \in I$. Define $s_{iu} = 0$ if the size u cannot be generated at location i.

$S_i \subseteq K_i$ — Set of all raw stock sizes that are processed at facility location $i \in I$; define $S = \bigcup_{i \in I} S_i$.

x_{ijk} — Rate of shipment of units of size $k \in D_j$ from facility location $i \in I$ to demand location $j \in J$.

$y^i_{m,n}$ — Rate of generation at facility location $i \in I$, of units of size $(m-n)$ that are obtained by cutting size $m(\geq n)$ down to size n.

z_i — Binary variable to decide whether facility i should be operated or not; $z_i = 1$, if facility $i \in I$ is opened; $z_i = 0$, otherwise.

D.1.3 Abbreviation

AC — The average number of feasibility cuts added to the master problem at each iteration.

ITE — The total number of iterations required to add all feasibility cuts required by the algorithm.

DC — The total number of dominated feasibility cuts added to the master problem before reaching optimality.

TC — The total number of feasibility cuts generated and added to the master problem before reaching optimality.

TIM — The total time (in seconds) to solve the problem.

D.2 NEW NOTATION FOR STOCHASTIC PROBLEM

D.2.1 Greek Letters

$\delta_i(\xi)$ — Cost per cut at facility $i \in I$ under scenario $\xi \in \Xi$; $\delta_i > 0$.

$\lambda_{jk}(\xi)$ — Demand rate at location $j \in J$ for size $k \in D_j(\xi)$ under scenario ξ.

$\xi \in \Xi$ — Possible scenario and Ξ is the set of all scenarios.

$\sigma_i(\xi)$ — Unit salvage value of product at facility $i \in I$ under scenario $\xi \in \Xi$; $\sigma_i > 0$.

D.2.2 Roman Letters

$a_{ih}(\xi)$ — Cost of one unit of raw stock of size $h \in S_i(\xi)$ at facility location $i \in I$ under scenario $\xi \in \Xi$.

$c_{ijk}(\xi)$ — Cost of transporting a single unit of size $k \in D_j(\xi)$ from facility location $i \in I$ to demand location $j \in J$ under scenario $\xi \in \Xi$.

$D_j(\xi)$ — Set of all sizes that are demanded at demand location $j \in J$ under scenario ξ; define $D = \bigcup_{j \in J, \xi \in \Xi} D_j(\xi)$.

$K_i(\xi)$ — Set of all possible sizes, including scrap sizes, that could be generated at facility $i \in I$ under scenario ξ (we include the size 0 in this set); define $K = \bigcup_{i \in I, \xi \in \Xi} K_i(\xi)$.

$M_i(\xi)$ — Total open production capacity for facility $i \in I$ per unit time under scenario $\xi \in \Xi$.

$p(\xi)$ — Probability of scenario ξ.

$r_{ih}(\xi)$ — Rate of replenishment of raw units of size $h \in S_i(\xi)$ at facility location $i \in I$ under scenario $\xi \in \Xi$; define $r_{ih}(\xi) = 0$ if $h \in (K_i(\xi) \backslash S_i(\xi))$ under scenario $\xi \in \Xi$.

$s_{iu}(\xi)$ — Rate at which size u is scrapped at facility location $i \in I$; define $s_{iu}(\xi) = 0$ if the size u cannot be generated at location i under scenario $\xi \in \Xi$.

$S_i(\xi)$ — Set of all raw stock sizes that are processed at facility location $i \in I$ under scenario ξ; define $S = \bigcup_{i \in I, \xi \in \Xi} S_i(\xi)$.

$x_{ijk}(\xi)$ — Rate of shipment of units of size $k \in D_j(\xi)$ from facility location $i \in I$ to demand location $j \in J$ under scenario $\xi \in \Xi$.

$y^i_{m,n}(\xi)$ — Rate of generation at facility location $i \in I$ of units of size $(m - n)$ that are obtained by cutting size m down to size n under scenario $\xi \in \Xi$.

D.2.3 Abbreviation

FC — The total number of feasibility cuts generated and added to the master problem before reaching optimality.

FIT — The total number of iterations required to add all feasibility cuts required by the algorithm.

OC — The total number of optimality cuts generated and added to the master problem before reaching optimality.

OIT — The total number of iterations required to add all optimality cuts required by one algorithm.

BIBLIOGRAPHY

[1] K. Aardal, Y. Pochet, and L. A. Wolsey. Capacitated facility location: Valid inequalities and facets. *Mathematics of Operations Research*, 20(3):562–582, 1995.

[2] D. Adelman and G. L. Nemhauser. Price-directed control of remnant inventory systems. *Operations Research*, 47(6):889–898, 1999.

[3] D. Adelman, G. L. Nemhauser, M. Padron, R. Stubbs, and R. Pandit. Allocating fibers in cable manufacturing. *Manufacturing and Service Operations Management*, 1(1):21–35, 1999.

[4] A. A. Ageev. Improved approximation algorithms for multilevel facility location problems. *Operations Research Letters*, 30(5):327–332, 2002.

[5] R. K. Ahuja, J. B. Orlin, S. Pallottino, M. P. Scaparra, and M. G. Scutellà. A multi-exchange heuristic for the single-source capacitated facility location problem. *Management Science*, 50(6):749–760, 2004.

[6] C. Aikens. Facility location models for distribution planning. *European Journal of Operational Research*, 22:263–279, 1985.

[7] U. Akinc and B. Khumawala. An efficient branch and bound algorithm for the capacitated warehouse location problem. *Management Science*, 23(6):585–594, 1997.

[8] J. Balakrishnan. An improved algorithm for solving a multi-period facility location problem. *IIE Transactions*, 36(1):19–22, 2004.

[9] R. Batta and N. R. Mannur. Covering-location models for emergency situations that require multiple response units. *Management Science*, 36(1):16–23, 1990.

[10] L. Bertazzi and M. G. Speranza. Minimizing logistics costs in multistage supply chains. *Naval Research Logistics*, 46(4):399–417, 1999.

[11] J. R. Birge and F. Louveaux. *Introduction to Stochastic Programming.* Springer, New York, 1997.

[12] J. R. Birge and F. V. Louveaux. A multicut algorithm for two-stage stochastic linear programs. *European Journal of Operational Research*, 34:384–392, 1988.

[13] J. R. Birge and L. Qi. Computing block-angular karmarkar projections with applications to stochastic programming. *Management Science*, 34(12):1472–1479, 1988.

[14] D. E. Blumenfeld, L. D. Burns, and C. F. Daganzo. Synchronizing production and transportation schedules. *Transportation Research*, 25B(1):23–37, 1991.

[15] D. E. Blumenfeld, L. D. Burns, J. D. Diltz, and C. F. Daganzo. Analyzing trade-offs between transportation, inventory and production costs on freight networks. *Transportation Research*, 19B(5):361–380, 1985.

[16] T. Boyaci and G. Gallego. Serial production/distribution systems under service constraints. *Manufacturing and Service Operations Management*, 3(1):43–50, 2001.

[17] J. Bramel, S. Goyal, and P. Zipkin. Coordination of production/distribution networks with unbalanced leadtimes. *Operations Research*, 48(4):570–577, 2000.

[18] D. Bredström, J. T. Lundgren, M. Rönnqvist, D. Carlsson, and A. Mason. Supply chain optimization in the pulp mill industry-IP models, column generation and novel constraint branches. *European Journal of Operational Research*, 156(1):2–22, 2004.

[19] M. J. Canós, C. Ivorra, and V. Liern. The fuzzy p-median problem: A global analysis of the solutions. *European Journal of Operational Research*, 130(2):430–436, 2001.

[20] L. M. A. Chan, A. Muriel, and D. Simchi-Levi. Supply-chain management: Integrating inventory and transportation. *Working Paper, Department of Industrial Engineering and Management Sciences, Northwestern University*, 98:1–18, 1997.

[21] C. Chen, S. Hart, and W. Tham. A simulated annealing heuristic for the one-dimensional cutting stock problem. *European Journal of Operational Research*, 93(3):522–535, 1996.

[22] M. Chen and W. Wang. A linear programming model for integrated steel production and distribution planning. *International Journal of Operations and Production Management*, 17(5/6):592–610, 1997.

[23] C. H. Cheng, B. R. Feiring, and T. C. E. Cheng. The cutting stock problem – a survey. *International Journal of Production Economics*, 36(3):291–305, 1994.

[24] D. C. Cho, E. L. Johnson, M. W. Padberg, and M. R. Rao. On the uncapacitated plant location problem I: Valid inequalities. *Mathematics of Operations Research*, 8:579–589, 1983.

[25] D. C. Cho, E. L. Johnson, M. W. Padberg, and M. R. Rao. On the uncapacitated plant location problem II: Facets and lifting theorems. *Mathematics of Operations Research*, 8:590–612, 1983.

[26] C. Chu and J. Antonio. Approximation algorithm to solve real-life multicriteria cutting stock problems. *Operations Research*, 47(4):495–508, 1999.

[27] F. A. Chudak and D. P. Williamson. Improved approximation algorithms for capacitated facility location problems. *Mathematical Programming*, 102(2):207–222, 2005.

[28] G. Cornuejols, R. Sridharan, and J. M. Thizy. A comparison of heuristics and relaxations for the capacitated plant location problem. *European Journal of Operational Research*, 50(3):280–297, 1991.

[29] G. Cornuéjols and J. M. Thizy. Some facets of the simple plant location problem. *Mathematical Programming*, 23:50–74, 1982.

[30] J. Current and M. O'Kelly. Locating emergency warning sirens. *Decision Sciences*, 23(1):221–234, 1992.

[31] A. Dasci and V. Verter. A continuous model for production-distribution system design. *European Journal of Operational Research*, 129(2):287–298, 2001.

[32] M. S. Daskin. A maximum expected covering location model: Formulation, properties and heuristic solution. *Transportation Science*, 17(1):48–70, 1983.

[33] Q. Deng and D. Simchi-Levi. Valid inequalities, facets and computational experience for the capacitated concentrator location problem. *European Journal of Operational Research*, 50:280–297, 1991.

[34] B. Denton and D. Gupta. Strategic inventory deployment in the steel industry. *IIE Transactions*, 36(11):1083–1097, 2004.

[35] B. Denton, D. Gupta, and K. Jawahir. Managing increasing product variety at integrated steel mills. *Interfaces*, 33(2):41–53, 2003.

[36] H. Dyckhoff. A new linear programming approach to the cutting stock problem. *Operations Research*, 29(6):1092–1104, 1981.

[37] H. Dyckhoff. A typology of cutting and packing problems. *European Journal of Operational Research*, 44(2):145–159, 1990.

[38] D. Erlenkotter. Dual-based procedure for uncapacitated facility location. *Operations Research*, 26:992–1009, 1978.

[39] H. I. Gassmann. MSLIP: A computer code for the multistage stochastic linear programming problem. *mathematical programming*, 47:407 – 423, 1990.

[40] A. M. Geoffrion and G. W. Graves. Multicommodity distribution system design by Benders decomposition. *Management Science*, 20(5):822–844, 1974.

[41] P. C. Gilmore and R. E. Gomory. A linear programming approach to the cutting stock problem. *Operations Research*, 9(6):849–859, 1961.

[42] P. C. Gilmore and R. E. Gomory. Multistage cutting stock problems of two and more dimensions. *Operations Research*, 13(1):94–120, 1965.

[43] P. C. Gilmore and R. E. Gomory. The theory and computation of knapsack functions. *Operations Research*, 14(6):1045–1074, 1966.

[44] E. Gourdin, M. Labbe, and G. Laporte. The uncapacitated facility location problem with client matching. *Operations Research*, 48(5):671–685, 2000.

[45] M. Guignard. Fractional vertices, cuts and facets for the simple plant location problem. *Mathematical Programming*, 12:150–162, 1980.

[46] R. W. Haessler and P. E. Sweeney. Cutting stock problems and solution procedures. *European Journal of Operational Research*, 54(2):141–150, 1991.

[47] L. C. Hendry, K. K. Fok, and K. W. Shek. A cutting-stock and scheduling problem in the copper industry. *Journal of the Operational Research Society*, 47(1):38–47, 1996.

[48] J. Higle and S. Sen. Stochastic decomposition: An algorithm for two stage linear programs with recourse. *Mathematics of Operations Research*, 16(3):650–669, 1991.

[49] R. Hinterding and K. Juliff. *A genetic algorithm for stock cutting: An exploration of mapping schemes.* Victoria University of Technology, Victoria, Australia: Department of Computer and Mathematical Sciences, 1993.

[50] R. Hinterding and L. Khan. Genetic algorithms for cutting stock problems: With and without contiguity. *Lecture Notes in Computer Science*, 956:166–186, 1993.

[51] F. L. Hitchcock. The distribution of a product from several sources to numerous localities. *Journal of Mathematics and Physics*, 20:224–230, 1941.

[52] D. S. Hochbaum and S. P. Hong. On the complexity of the production-transportation problem. *SIAM Journal on Optimization*, 6(1):250–264, 1996.

[53] J. H. Holland. Genetic algorithms. *Scientific American*, 267(1):66–72, 1992.

[54] K. Holmberg and H. Tuy. A production-transportation problem with stochastic demand and concave production costs. *Mathematical Programming*, 85(1):157–179, 1999.

[55] S. K. Jacobsen. Heuristics for the capacitated plant location model. *European Journal of Operational Research*, 12:253–261, 1983.

[56] V. Jayaraman and H. Pirkul. Planning and coordination of production and distribution facilities for multiple commodities. *European Journal of Operational Research*, 133(2):394–408, 2001.

[57] M. P. Johnson, C. Rennick, and E. Zak. Skiving addition to the cutting stock problem in the paper industry. *SIAM Review*, 39(3):472–483, 1997.

[58] P. Kall and S. W. Wallace. *Stochastic Programming.* Wiley, New York, 1994.

[59] B. M. Khumawala. An efficient branch and bound algorithm for the capacitated warehouse location problem. *Management Science*, 18(12):B718–731, 1972.

[60] M. Korupolu, C. Plaxton, and R. Rajaraman. Analysis of a local search heuristic for facility location problems. *Journal of Algorithms*, 37:146–174, 2000.

[61] E. V. Krichagina, R. Rubio, M. I. Taksar, and L. M. Wein. A dynamic stochastic stock cutting problem. *Operations Research*, 46(5):690–701, 1998.

[62] T. Kuno and T. Utsunomiya. A lagrangian based branch-and-bound algorithm for production-transportation problems. *Journal of Global Optimization*, 18(1):59–73, 2000.

[63] C. Y. Lee. An optimal algorithm for the multiproduct capacitated facility location problem with a choice of facility type. *Computers and Operations Research*, 18:167–182, 1991.

[64] C. Y. Lee. A cross decomposition algorithm for a multiproduct-multitype facility location problem. *Computers and Operations Research*, 20:527–540, 1993.

[65] J. M. Y. Leung and T. L. Magnanti. Valid inequalities and facets of the capacitated plant location problem. *Mathematical Programming*, 44:271–291, 1989.

[66] Y. Levin and A. Ben-Israel. A heuristic method for large-scale multi-facility location problems. *Computers and Operations Research*, 31(2):257–272, 2004.

[67] W. J. Li and J. M. Smith. An algorithm for quadratic assignment problems. *European Journal of Operational Research*, 81:205–216, 1995.

[68] K. Liang, X. Yao, C. Newton, and D. Hoffman. A new evolutionary approach to cutting stock problems with and without contiguity. *Computers and Operations Research*, 29(12):1641–1659, 2002.

[69] Y. Lirov. Knowledge based approach to the cutting stock problem. *Mathematical and Computer Modelling*, 16(1):107–125, 1992.

[70] L. A. N. Lorena and E. L. F. Senne. A column generation approach to capacitated p-median problems. *Computers and Operations Research*, 31(6):863–876, 2004.

[71] F. V. Louveaux. Stochastic location analysis. *Location Science*, 1:127–154, 1993.

[72] F. V. Louveaux and L. V. Hamme. Exact solution to a location problem with stochastic demands. *Transportation Science*, 28(2):95–103, 1994.

[73] F. V. Louveaux and D. Peeters. A dual-based procedure for stochastic facility location. *Operations Research*, 40(3):564–573, 1992.

[74] R. K. Martin. *Large Scale Linear and Integer Optimization: A unified approach.* Kluwer Academic Publishers, 1999.

[75] J. B. Mazzola and A. W. Neebe. Lagrangian-relaxation-based solution procedures for a multiproduct capacitated facility location problem with choice of facility type. *European Journal of Operational Research*, 115:285–299, 1999.

[76] P. A. Miranda and R. A. Garrido. Incorporating inventory control decisions into a strategic distribution network design model with stochastic demand. *Transportation Research Part E*, 40:183207, 2004.

[77] P. B. Mirchandani and R. L. Francis. *Discrete Location Theory.* John Wiley and Sons, Inc., New York, USA, 1990.

[78] R. P. Mohanty and R. Singh. A hierarchical production planning approach for a steel manufacturing system. *International Journal of Operations and Production Management*, 12(5):69–78, 1992.

[79] S. Mukundan and M. S. Daskin. Joint location/sizing maximum profit covering models. *INFOR*, 29(2):139–152, 1991.

[80] North Carolina Department of Environment and Natural Resources (NCDENR). Metals: Aluminum cans and scrap commodity profile/markets assessment. *http://www.p2pays.org/ref/02/0162209.pdf*, pages 1–8, 1998.

[81] S. H. Owen and M. S. Daskin. Strategic facility location: A review. *European Journal of Operational Research*, 111(3):423–447, 1998.

[82] E. C. Partington. Strategic inventory management of renewable energy resources: Aluminum as an energy storage medium and market commodity. *International Journal of Production Economics*, 26(1-3):211–216, 1992.

[83] H. Pirkul and R. Gupta. Visopt: A visual interactive optimization tool for p-median problems. *Decision Support Systems*, 26(3):209–223, 1999.

[84] Ivan Png. *Managerial Economics.* Blackwell Publishers Inc., 350 Main Street, Malden, Massachusetts, 02148 USA, 2002.

[85] D. F. Pyke and M. A. Cohen. Multiproduct integrated production-distribution systems. *European Journal of Operational Research*, 74(1):18–49, 1994.

[86] S. Robertson. Out of control. *American Metal Market*, 112(31):4–5, 2004.

[87] K. E. Rosing, C. S. Revelle, and D. A. Schilling. A gamma heuristic for the p-median problem. *European Journal of Operational Research*, 117(3):522–532, 1999.

[88] B. Sasidhar and K. K. Achary. A multiple arc network model of production planning in a steel mill. *International Journal of Production Economics*, 22(3):195–202, 1991.

[89] L. A. Schaefer and A. J. Schaefer. Locating hybrid fuel cell - turbine power generation units under uncertainty. *Annals of Operations Research*, 132(1-4):301 –322, 2004.

[90] G. Scheithauer, J. Terno, A. Muller, and G. Belov. Solving one-dimensional cutting stock problems exactly with a cutting plane algorithm. *Journal of the Operational Research Society*, 52(12):1390–1401, 2001.

[91] Z. M. Shen, C. Coullard, and M. S. Daskin. A joint location-inventory model. *Transportation Science*, 37(1):40–55, 2003.

[92] R. Sherefkin. After respite, steel prices soar. *Automotive News*, 78(6105):8–9, 2004.

[93] R. V. Slyke and R. J-B Wets. L-shaped linear programs with application to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17:638–663, 1969.

[94] J. C. Smith, A. J. Schaefer, and J. W. Yen. A stochastic integer programming approach to solving a synchronous optical network ring design problem. *Networks*, 44(1):12–26, 2004.

[95] M. G. Speranza and W. Ukovich. Minimizing transportation and inventory costs for several products on a single link. *Operations Research*, 42(5):879–894, 1994.

[96] R. Sridharan. The capacitated plant location problem. *European Journal of Operational Research*, 87:203–213, 1995.

[97] P. E. Sweeney and E. R. Paternoster. Cutting and packing problems: A categorized, application-orientated research bibliography. *Journal of the Operational Research Society*, 43(7):691–706, 1992.

[98] L. Tang, J. Liu, A. Rong, and Z. Yang. A review of planning and scheduling systems and methods for integrated steel production. *European Journal of Operational Research*, 133(10):1–20, 2001.

[99] L. Tang, P. B. Luh, J. Liu, and L. Fang. Steel-making process scheduling using lagrangian relaxation. *International Journal of Production Research*, 40(1):55–70, 2002.

[100] C. Toregas, R. Swain, C. ReVelle, and L. Bergman. The location of emergency service facilities. *Operations Research*, 19(6):1363–1373, 1971.

[101] H. Tuy, N. D. Dan, and S. Ghannadan. Strongly polynomial time algorithms for certain concave minimization problems on networks. *Operations Research Letters*, 14(2):99–109, 1993.

[102] H. Tuy, S. Ghannadan, A. Migdalas, and P. Värbrand. A strongly polynomial algorithm for a concave production-transportation problem with a fixed number of nonlinear variables. *Mathematical Programming, Series B*, 72(3):229–258, 1996.

[103] J. M. Valério de Carvalho. Exact solution of cutting stock problems using column generation and branch-and-bound. *International Transactions in Operational Research*, 5(1):35–44, 1998.

[104] J. M. Valério de Carvalho. Lp models for bin packing and cutting stock problems. *European Journal of Operational Research*, 141(2):253–273, 2002.

[105] M. Van Buer, D. Woodruff, and R. T. Olson. Solving the medium newspaper production/distribution problem. *European Journal of Operational Research*, 115(2):237–253, 1999.

[106] T. J. Van Roy. A cross decomposition algorithm for capacitated facility location. *Operations Research*, 34(1):145–163, 1986.

[107] R. M. Van Slyke and R. J. B. Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal of Applied Mathematics*, 17(4):638–663, 1969.

[108] J. L. G. Velarde and M. Laguna. A Benders-based heuristic for the robust capacitated international sourcing problem. *IIE Transactions*, 36:1125–1133, 2004.

[109] C. J. Vidal and M. Goetschalckx. Strategic production-distribution models: A critical review with emphasis on global supply chain models. *European Journal of Operational Research*, 98:1–18, 1997.

[110] B. J. Wagner. A genetic algorithm solution for one-dimensional bundled stock cutting. *European Journal of Operational Research*, 117(2):368–381, 1999.

[111] Q. Wang, R. Batta, and C. M. Rump. Algorithms for a facility location problem with stochastic customer demand and immobile servers. *Annals of Operations Research*, 111:17–34, 2002.

[112] P. Wentges. Accelerating Benders' decomposition for the capacitated facility location problem. *Mathematical Methods of Operations Research*, 44(2):267–290, 1996.

[113] J. F. Williams. A hybrid algorithm for simultaneous scheduling of production and distribution in multi-echelon structures. *Management Science*, 29(1):77–92, 1983.

[114] R. M. Wollmer. Two-stage linear programming under uncertainty with 0-1 first stage variables. *Mathematical Programming*, 19:279–288, 1980.

[115] S. Zanoni and L. Zavanella. Model and analysis of integrated production-inventory system: The case of steel production. *International Journal of Production Economics*, 93/94:197–205, 2005.

[116] J. Zhou and B. Liu. New stochastic models for capacitated location-allocation problem. *Computers and Industrial Engineering*, 45:111–125, 2003.