RESOURCE CONSTRAINED ASSIGNMENT PROBLEMS WITH FLEXIBLE CUSTOMER DEMAND

By

CHASE E. RAINWATER

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2009
To my wife, my parents and my brother
ACKNOWLEDGMENTS

First, to my beautiful wife. This dissertation represents another milestone in our adventure together. For seeing me through the frustrations and trials that undoubtedly accompanied the development of this dissertation, I am forever grateful. Having an unwavering source of happiness to go home to is the secret to my success. To that end, I share any success that I’ve had with you.

To my family, your love and support have been invaluable. Specifically to my parents, thank you for all you did to prepare for this experience.

To Joe Geunes and Edwin Romeijn, I cannot begin to describe the impact you have had on me academically, professionally and personally. The chance to work for you ranks as one of the greatest privileges of my life. I owe the opportunity to pursue an academic career to the two of you. I only hope that I can be half the mentor to others, as you’ve been to me.

To Cole Smith, I thank you for being such an unbelievable teacher and mentor throughout graduate school. Also, thank you for the dog.

Lastly, to Caner, Semra and the rest of my graduate student colleagues, thank you for your friendship throughout these four years. I remain amazed not only by your talents and abilities, but, more importantly, by the sincere kindness you’ve consistently shown me.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>4</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>8</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>11</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>12</td>
</tr>
<tr>
<td>CHAPTER</td>
<td></td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>14</td>
</tr>
<tr>
<td>2 LITERATURE REVIEW</td>
<td>19</td>
</tr>
<tr>
<td>2.1 Capacitated Facility Location Problem with Single-Source Constraints</td>
<td>19</td>
</tr>
<tr>
<td>2.2 Generalized Assignment Problem</td>
<td>22</td>
</tr>
<tr>
<td>2.3 0-1 Knapsack Problem</td>
<td>26</td>
</tr>
<tr>
<td>2.4 Flexible Demand Assignment Problems</td>
<td>28</td>
</tr>
<tr>
<td>3 CAPACITATED FACILITY LOCATION WITH SINGLE-SOURCE CONSTRAINTS AND FLEXIBLE DEMAND</td>
<td>30</td>
</tr>
<tr>
<td>4 EXACT ALGORITHM FOR CFLFD AND GAPFD</td>
<td>35</td>
</tr>
<tr>
<td>4.1 Alternative Representation of the CFLFD</td>
<td>36</td>
</tr>
<tr>
<td>4.1.1 Set-Partitioning Formulation</td>
<td>36</td>
</tr>
<tr>
<td>4.1.2 Pricing Problem</td>
<td>38</td>
</tr>
<tr>
<td>4.2 Knapsack Problem with Expandable Items</td>
<td>39</td>
</tr>
<tr>
<td>4.2.1 CFLFD with Specially Structured Revenue Functions</td>
<td>43</td>
</tr>
<tr>
<td>4.2.2 Convex and Linear Revenue Functions</td>
<td>44</td>
</tr>
<tr>
<td>4.3 Branch-and-Price Algorithm Implementation</td>
<td>48</td>
</tr>
<tr>
<td>4.3.1 Initial Feasible Solution</td>
<td>48</td>
</tr>
<tr>
<td>4.3.2 Solving LP(RSP)</td>
<td>49</td>
</tr>
<tr>
<td>4.3.3 Node and Variable Selection</td>
<td>51</td>
</tr>
<tr>
<td>4.3.4 Optimal Column Cost</td>
<td>52</td>
</tr>
<tr>
<td>4.4 Computational Results</td>
<td>53</td>
</tr>
<tr>
<td>4.4.1 Experimental Data</td>
<td>53</td>
</tr>
<tr>
<td>4.4.2 CFLFD Results</td>
<td>56</td>
</tr>
<tr>
<td>4.4.2.1 Nonlinear revenue functions: comparison with BARON</td>
<td>56</td>
</tr>
<tr>
<td>4.4.2.2 Piecewise linear and linear revenue functions: comparison with CPLEX</td>
<td>58</td>
</tr>
<tr>
<td>4.4.3 GAPFJ Results</td>
<td>59</td>
</tr>
<tr>
<td>4.5 Conclusions</td>
<td>61</td>
</tr>
</tbody>
</table>
7.7.1 Initial Feasible Solution ................................................. 164
7.7.2 Heuristics for PP\textsubscript{i}-F .............................................. 165
7.7.3 Solving LP(RMP-F) ........................................................ 167
7.7.4 Node and Variable Selection ........................................... 168
7.8 Computational study ......................................................... 169
7.8.1 Experimental Design ...................................................... 169
7.8.2 Base Results .............................................................. 171
7.8.3 Extended Results ......................................................... 173
7.9 Conclusions and Future Research ....................................... 175

8 CONCLUSION .............................................................................. 184

APPENDIX
A GAPFD ASYMPTOTIC PROPERTY ............................................. 187
B CFLFD PRICING PROBLEM PROPERTY .................................... 189
REFERENCES ............................................................................ 191
BIOGRAPHICAL SKETCH ............................................................. 198
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-1</td>
<td>CFLFD with nonlinear revenue functions: 5 facilities, 10 customers, $\tau = 1.2$</td>
<td>61</td>
</tr>
<tr>
<td>4-2</td>
<td>CFLFD with nonlinear revenue functions: 5 facilities, 15 customers, $\tau = 1.2$</td>
<td>63</td>
</tr>
<tr>
<td>4-3</td>
<td>CFLFD with nonlinear revenue functions: 5 facilities, 25 customers, $\tau = 1.2$</td>
<td>64</td>
</tr>
<tr>
<td>4-4</td>
<td>Branching rule comparison for the CFLFD with nonlinear revenue functions: 5 facilities, 10 customers, $\tau = 1.2$</td>
<td>64</td>
</tr>
<tr>
<td>4-5</td>
<td>CFLFD with piecewise linear convex revenue functions: 5 facilities, 10 customers, $\tau = 1.2$</td>
<td>65</td>
</tr>
<tr>
<td>4-6</td>
<td>CFLFD with piecewise linear convex revenue functions: 5 facilities, 15 customers, $\tau = 1.2$</td>
<td>65</td>
</tr>
<tr>
<td>4-7</td>
<td>CFLFD with piecewise linear convex revenue functions: 5 facilities, 25 customers, $\tau = 1.2$</td>
<td>66</td>
</tr>
<tr>
<td>4-8</td>
<td>CFLFD with piecewise linear concave revenue functions: 5 facilities, 10 customers, $\tau = 1.2$</td>
<td>66</td>
</tr>
<tr>
<td>4-9</td>
<td>CFLFD with piecewise linear concave revenue functions: 5 facilities, 15 customers, $\tau = 1.2$</td>
<td>67</td>
</tr>
<tr>
<td>4-10</td>
<td>CFLFD with piecewise linear concave revenue functions: 5 facilities, 25 customers, $\tau = 1.2$</td>
<td>67</td>
</tr>
<tr>
<td>4-11</td>
<td>CFLFD with linear revenue functions: 30 facilities, 60 customers, $\tau = 1.2$</td>
<td>68</td>
</tr>
<tr>
<td>4-12</td>
<td>GAPFJ with nonlinear revenue functions: 5 facilities, 10 customers, $\tau = 1.2$</td>
<td>68</td>
</tr>
<tr>
<td>4-13</td>
<td>GAPFJ with nonlinear revenue functions: 5 facilities, 15 customers, $\tau = 1.2$</td>
<td>69</td>
</tr>
<tr>
<td>4-14</td>
<td>GAPFJ with nonlinear revenue functions: 5 facilities, 25 customers, $\tau = 1.2$</td>
<td>69</td>
</tr>
<tr>
<td>4-15</td>
<td>GAPFJ with linear revenue functions: 30 facilities, 90 customers, $\tau = 1.2$</td>
<td>70</td>
</tr>
<tr>
<td>5-1</td>
<td>Facility-independent requirements: 15 facilities, $\tau = 1.1$</td>
<td>105</td>
</tr>
<tr>
<td>5-2</td>
<td>Facility-independent requirements: 15 facilities, $\tau = 1.2$</td>
<td>105</td>
</tr>
<tr>
<td>5-3</td>
<td>Facility-independent requirements: 15 facilities $\tau = 1.3$</td>
<td>105</td>
</tr>
<tr>
<td>5-4</td>
<td>Facility-independent requirements: 30 facilities, $\tau = 1.1$</td>
<td>105</td>
</tr>
<tr>
<td>5-5</td>
<td>Facility-independent requirements: 30 facilities, $\tau = 1.2$</td>
<td>105</td>
</tr>
<tr>
<td>5-6</td>
<td>Facility-independent requirements: 30 facilities, $\tau = 1.3$</td>
<td>106</td>
</tr>
</tbody>
</table>
5-7 Facility-dependent requirements: 15 facilities, $\tau = 1.1$ ........................................ 106
5-8 Facility-dependent requirements: 15 facilities, $\tau = 1.2$ ........................................ 106
5-9 Facility-dependent requirements: 15 facilities, $\tau = 1.3$ ........................................ 106
5-10 Facility-dependent requirements: 30 facilities, $\tau = 1.1$ ........................................ 106
5-11 Facility-dependent requirements: 30 facilities, $\tau = 1.2$ ........................................ 107
5-12 Facility-dependent requirements: 30 facilities, $\tau = 1.3$ ........................................ 107
5-13 Post-processing effect on heuristic with greedy and improvement phase; facility-independent
requirements: $\tau = 1.2$ ........................................ 107
5-14 Post-processing effect; facility-dependent requirements: $\tau = 1.2$ ................................ 107
6-1 FNS results: 15 facilities ........................................ 129
6-2 RNS results: 15 facilities ........................................ 129
6-3 FNS results: 30 facilities ........................................ 129
6-4 RNS results: 30 facilities ........................................ 130
7-1 FASR: 15 facilities, 30 customers, 3 customer types, $\tau_a = \tau_t = 1.2$, $\phi_i=.5$ ($i \in \mathcal{I}$), $\Theta = 5$ ........................................ 178
7-2 FASR: 15 facilities, 45 customers, 3 customer types, $\tau_a = \tau_t = 1.2$, $\phi_i=.5$ ($i \in \mathcal{I}$), $\Theta = 5$ ........................................ 178
7-3 FASR: 30 facilities, 60 customers, 3 customer types, $\tau_a = \tau_t = 1.2$, $\phi_i=.5$ ($i \in \mathcal{I}$), $\Theta = 5$ ........................................ 179
7-4 FASR: 30 facilities, 90 customers, 3 customer types, $\tau_a = \tau_t = 1.2$, $\phi_i=.5$ ($i \in \mathcal{I}$), $\Theta = 25$ ........................................ 180
7-5 FASR: 15 facilities, 45 customers, 3 customer types, $\tau_a = \tau_t = 1.2$, $\phi_i=.9$ ($i \in \mathcal{I}$), $\Theta = 5$ ........................................ 181
7-6 FASR: 15 facilities, 30 customers, 3 customer types, $\tau_a = \tau_t = 1.2$, $\phi_i=.2$ ($i \in \mathcal{I}$), $\Theta = 5$ ........................................ 180
7-7 FASR: 15 facilities, 45 customers, 3 customer types, $\tau_a = \tau_t = 1.2$, $\phi_i=.2$ ($i \in \mathcal{I}$), $\Theta = 5$ ........................................ 181
7-8 FASR: 15 facilities, 75 customers, 3 customer types, $\tau_a = \tau_t = 1.2$, $\phi_i=.9$ ($i \in \mathcal{I}$), $\Theta = 5$ ........................................ 181
7-9 FASR: 15 facilities, 30 customers, 3 customer types, $\tau_a = 1.2$, $\tau_t = 1.1$, $\phi_i=.5$
($i \in \mathcal{I}$), $\Theta = 5$ ........................................ 182
7-10 FASR: 15 facilities, 45 customers, 3 customer types, $\tau_a = 1.2$, $\tau_t = 1.1$, $\phi_i = .5$
$(i \in I)$, $\Theta = 5$ .................................................. 182

7-11 FASR: 15 facilities, 30 customers, 3 customer types, $\tau_a = 1.2$, $\phi_i = .5$ $(i \in I)$,
$\Theta = 5$, $g_q = \infty$ $(q \in Q)$ .................................................. 183

7-12 FASR: 15 facilities, 45 customers, 3 customer types, $\tau_a = 1.2$, $\phi_i = .5$ $(i \in I)$,
$\Theta = 5$, $g_q = \infty$ $(q \in Q)$ .................................................. 183
### LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4-1</td>
<td>Illustration of $\phi_j$ and $\theta_j$ for a general revenue function</td>
<td>62</td>
</tr>
<tr>
<td>4-2</td>
<td>Concave envelope: convex revenue function</td>
<td>62</td>
</tr>
<tr>
<td>4-3</td>
<td>Concave envelope: linear revenue function</td>
<td>63</td>
</tr>
<tr>
<td>7-1</td>
<td>Illustration of $r_q$ and $\theta_q$</td>
<td>177</td>
</tr>
</tbody>
</table>
This dissertation considers classes of problems that seek to make profitable demand fulfillment decisions with limited available resources. This general problem scenario has been given much consideration over recent decades. In this work, we add to this body of research by considering less explored problem variants that allow decision makers to exploit demand flexibility to increase profit. We first consider a generalization of the capacitated facility location with single-sourcing constraints. Each customer must be assigned to a procured facility, and the level at which the customer’s demand is fulfilled (a decision variable) must be determined, subject to falling within pre-specified limits. A customer’s revenue is nondecreasing in its resource consumption, according to a general revenue function, and a fixed cost is incurred for each resource procured. We provide an exact branch-and-price algorithm that solves both this problem and a special case in which resource procurement is not considered. Our approach identifies an equally interesting class of pricing subproblems. We discuss how this class of problems can be solved with generalized revenue functions and offer efficient algorithms for solving instances with specially structured revenue functions that correspond to common pricing structures. Our extensive computational study compares the performance of our exact algorithm to that of well-known commercial solvers and demonstrates the advantages of our algorithmic approach for various categories of problem instances. Since
real-world scenarios often result in large-scale problem sizes, we consider novel heuristic approaches for both the generalization of the capacitated facility location problem and a particular special case, which can be viewed as an extension of the well-known Generalized Assignment Problem (GAP). We first develop a class of heuristic solution methods for the variant without resource procurement decisions. Our approach is motivated by a rigorous study of the linear relaxation of the model. We show that our class of heuristics is asymptotic optimality in a probabilistic sense under a broad stochastic model. Improvement procedures are discussed and a thorough computational study confirms our theoretical results. We then provide fast and practically implementable optimization-based heuristic solution methods for the generalized class of facility location problems with resource procurement decisions. Our procedure is designed for very large-scale problem instances. We offer a unique approach that utilizes a high-quality efficient heuristic within a neighborhood search to address the combined assignment and fixed-charge structure of the underlying optimization problem. We also study the potential benefits of combining our approach with a so-called very large-scale neighborhood search (VLSN) method. As our computational test results indicate, our work offers an attractive solution approach that can be tailored to successfully solve a broad class of problem instances for facility location and similar fixed-charge problems. Finally, we consider a separate class of assignment problems with non-linear resource consumption and non-traditional capacity constraints. The model is applicable to manufacturing scenarios in which products with common production characteristics share setup times or some element of fixed resource consumption. The additional capacity constraints account for real-world restrictions that may result from environmental guidelines, transportation resource limitations, or limited warehouse storage space. We propose a branch-and-price algorithm for this class of problems that requires a unique reformulation of our problem, as well as a study of a new class of knapsack problems. A computational study demonstrates the appeal of our approach over commercial solvers for various problem instances.
CHAPTER 1
INTRODUCTION

Optimization models which determine the most profitable manner to fulfill customer demand have been widely explored in the operations research literature for over 50 years. Many of the classical problems studied in our field, such as the Assignment Problem (Kuhn [55]), the Generalized Assignment problem (Ross and Soland [84]), the Capacitated Facility Location Problem (Nauss [71]), the Traveling Salesman Problem (Lin and Kernighan [61]), the Vehicle Routing Problem (Laporte [58]), and the numerous Fixed-Charge Transportation Problems (Adlakha and Kowalski [1]) have considered the assignment of customers to resources under varying real-world scenarios. More recently, the operations literature has emphasized ways to exploit sources of supply and demand flexibility to increase profit margins through demand and revenue management (see, e.g., Talluri and Van Ryzin [94]). This has led to a number of new models that focus on profit maximization by accounting for both the costs and revenue implications associated with operations decisions. For example, Chen and Hall [22] introduce several new “maximum profit scheduling” models that implicitly account for the fact that operations scheduling decisions can affect demand and therefore revenue. Another stream of literature considers optimal inventory management when demand levels (and hence revenues) depend on inventory levels (e.g., Baker and Urban [11, 12], Gerchak and Wang [43], and Balakrishnan et al. [14]) and/or shelf space allocation (Wang and Gerchak [99]), both of which impact operations costs. Several papers have also considered maximizing profit in production planning contexts with price-dependent demand, where production and inventory costs are determined by solving an optimization problem containing a lot-sizing structure (e.g., Thomas [95], Kunreuther and Schrage [56], Gilbert [46], Biller et al. [18], Deng and Yano [28], Geunes, Romeijn, and Taaffe [44], and van den Heuvel and Wagelmans [98]).

The work in this dissertation adds to this broad body of research. In the most general terms, we consider a set of customers, each with their own demand requirements, and
a set of resource constrained facilities. Our models seek to assign each customer to a single capacitated facility (resource) in a manner which maximizes total profit. As we will discuss in Chapter 2, this standard resource constrained assignment problem is itself difficult to solve, and numerous exact and heuristic solution methodologies have already been proposed. The work in this dissertation explores variants of this traditional problem which account for unique planning characteristics that may be available to a decision maker. Specifically, we first consider an extension in which the model allows for so-called flexible customer demand. This flexibility allows a supplier to better match customer demands with operations resources, therefore increasing profits. Real-world scenarios that allow for demand flexibility are often found in the production of construction materials, such as steel and wood. In these environments, distributors of these materials will accept deliveries from suppliers in a range of sizes (Balakrishnan and Geunes [13]). The distributors permit this flexibility because they often perform further customized cutting and finishing operations for their own customers, whose exact size specifications are not known to the distributor in advance. Suppliers to such distributors are often compensated based on total weight delivered to the distributor (within certain limits deemed acceptable to the distributor). Clearly, if the supplier has unlimited resources, they can maximize profit by delivering at the upper limit of the distributor’s stated acceptable size range. If, however, the supplier faces resource constraints (e.g., in terms of its quantity and sizes of raw materials) and must meet each element of a collection of customer demands, the problem of assigning these demands to available resources in order to maximize net profit is non-trivial. The first model considered in this dissertation requires that the facilities utilized to fulfill demand must be determined by the decisions maker. We refer to this problem as the Capacitated Facility Location Problem with Single-Source Constraints and Flexible Demand (CFLFD). In addition, we study the special case of CFLFD in which demand is fulfilled by a fixed set of facilities (i.e. procurement decisions are omitted).
We refer to this problem as the *Generalized Assignment Problem with Flexible Demand* (GAPFD). For each problem, we propose both exact and heuristic methodologies.

The exact approach utilized for the CFLFD and the GAPFD reformulates the problem as a set-partitioning problem. The resulting subproblem to be solved takes the form of an interesting class of non-linear knapsack problems. We derive structural results for an important relaxation of this class of knapsack problems. These results suggest efficient heuristic and exact approaches for solving these knapsack problems with various revenue functions. By considering these alternative revenue structures, our model accounts for quantity discounts and economies of scale, as well as revenues for specialized goods. Lastly, we provide a detailed discussion of how the customer demand fulfillment levels are determined in this approach, as well as how difficulties that arise when considering facility procurement can be overcome.

Solving large-scale instances of assignment-based problems is an issue that is actively considered in the literature. Therefore, a portion of this dissertation focuses on heuristic procedures that can be used to solve real-world size instances. While the exact approach for the flexible demand problem studied is applicable to problems with and without facility procurement, our heuristic approaches require separate consideration. For the GAPFD, we pursue a greedy algorithm that is motivated by an analysis of the linear relaxation of our model. Importantly, the heuristic presented is shown to have asymptotic performance guarantees under a very general stochastic model. For very large problems, the computational results suggest that the heuristic produces near-optimal solutions in dramatically reduced time when compared to that required by well-known commercial solvers.

As we discuss in detail throughout the dissertation, fixed-charge heuristics require careful design to assure that the fixed-charge decisions are fully considered. Therefore, for the CFLFD, we develop a heuristic framework that searches a separate facility neighborhood, while relying on constructive heuristics to determine the customer
assignments and demand fulfillment levels. We present the computational savings of this approach over other large-scale search heuristics and discuss the potential impacts that the framework might have on a number of other fixed-charge problems.

Clearly the consideration of flexible demand has notable practical significance. Moreover, solving problems with this element is a challenge that requires novel solution approaches. However, in real-world production environments, the total resources required to satisfy customer demand (flexible, or otherwise) may not be limited to the cumulative capacity consumed by the individual customers. As we will discuss, more often, certain production requirements are shared amongst subsets of customers. To model this, we allow for each customer to belong to a single type. Then, customers of each type consume a shared amount of resource in addition to the individual consumption required to fulfill demand. In addition to this consideration, we are also interested in different forms of capacity constraints. Typically, capacitated assignment problems consider only the capacity limitations of individual facilities. As we will establish in the review of literature in this area, assignment problems with capacity restrictions on customers of a particular type (regardless of which facility they are assigned) have received much less attention. Accounting for both of these problem elements requires a separate model from those that have been previously developed. While assignment problems with these additional characteristics can still be viewed as determining a feasible partition of customers amongst a set of available resources (facilities), the impact of the additional capacity constraints requires special consideration in the exact algorithm that is proposed. The result is an effective algorithm for another difficult class of optimization problems with strong practical implications.

The remainder of the dissertation is organized as follows. In Chapter 2, we discuss extensively the relevant literature. In Chapter 3, we formally present the first class of problems to be studied and introduce the concept of flexible demand. Chapter 4 discusses an exact approach for the CFLFD and the GAFPD. A new class of knapsack problems
that arise in our approach is presented. Both a formal study of the structure of these
problems and efficient solution procedures are provided. In Chapter 5, a class of heuristics
is proposed for the GAPFD. A probabilistic analysis demonstrates that our class of
heuristics is asymptotically feasible and optimal under a very general stochastic model.
A thorough computational study to assess the heuristic’s performance is completed and
compared against the theoretical claims made in the chapter. In Chapter 6, a heuristic
framework for the CFLFD with linear revenue functions is presented. We develop a
neighborhood search heuristic that decomposes the CFLFD and solves the corresponding
subproblem with the heuristic proposed in Chapter 5. We discuss the benefits of our
approach versus those commonly applied for fixed-charge optimization problems. The
motivation behind the chosen implementation is provided along with lessons learned
with respect to less successful implementations. Then, in Chapter 7, a new class of
problems is introduced which accounts for shared resource consumption among sets of
customers, as well as additional capacity constraints. An exact approach is proposed based
on a set-partitioning formulation of the model with complicating capacity constraints.
Important differences between this reformulation and the reformulation presented in
Chapter 4 are discussed. Another interesting class of knapsack problems is studied and
efficient solution approaches presented. Finally, Chapter 8 offers concluding remarks, as
well as a discussion of future research.
CHAPTER 2
LITERATURE REVIEW

The problems studied in this dissertation share characteristics with numerous classical optimization problems. This chapter provides an extensive review of the literature related to each of these problems. The discussion of work in each area focuses on the exact and heuristic methods that have been pursued. The presentation of these successful approaches will serve to motivate each of the solution methods chosen throughout the remainder of the dissertation.

2.1 Capacitated Facility Location Problem with Single-Source Constraints

The problems studied in this dissertation can be viewed as generalizations of well-known optimization problems. Of particular relevance to the problems introduced in Chapter 3 and studied in Chapters 4 and 6 is the Capacitated Facility Location Problem with Single-Source Constraints (CFLP).

\[
\text{maximize } \sum_{i \in I} \sum_{j \in J} p_{ij} x_{ij} - \sum_{i \in I} f_i y_i \quad (2-1)
\]

subject to (CFLP)

\[
\sum_{j \in J} a_{ij} x_{ij} \leq b_i y_i \quad i \in I \quad (2-2)
\]

\[
\sum_{i \in I} x_{ij} = 1 \quad j \in J \quad (2-3)
\]

\[
x_{ij} \in \{0, 1\} \quad i \in I; j \in J \quad (2-4)
\]

\[
y_i \in \{0, 1\} \quad i \in I. \quad (2-5)
\]

In this model, demand for customer \(j (j \in J)\) must be satisfied by a single procured capacitated facility \(i (i \in I)\), as enforced in constraints (2–3). A customer \(j (j \in J)\), assigned to facility \((i \in I)\), results in a profit amount \(p_{ij}\) and consumes \(a_{ij}\) units of facility \(i\)’s capacity. Constraints (2–2) ensure that customer demand is executed solely by procured facilities and that each facility’s resource availability is satisfied. Constraints
(2–4) and (2–5) place binary restrictions on the assignment variables $x_{ij}$ ($i \in I; j \in J$) and facility procurement variables $y_i$ ($i \in I$).

The CFLP with single-source constraints falls into the difficult class of $\mathcal{NP}$-Hard optimization problems (see Garey and Johnson [39]), implying that it is unlikely that a polynomial-time solution method exists for solving problems in this class (unless $\mathcal{P} = \mathcal{NP}$). Note that this negative complexity result even holds for the classical CFLP in which each customer’s demand may be split between the acquired facilities as long as all customer demands are allocated. In practice, a single-sourcing restriction, which requires that any customer’s demand must be allocated in its entirety to exactly one of the open facilities, is often imposed for a variety of reasons. For example, in the facility location context, single sourcing reduces coordination complexity, reduces the number of deliveries required to a customer, ensures consistency of deliveries received by customers, and provides customers with a single point of contact for supply. Moreover, a customer’s demand may depend on the facility to which it is assigned. This generalization is particularly relevant if the “facilities” represent machines or people, each with different processing capabilities, and is therefore often found in production environments. With the single-sourcing constraint and facility-dependent demands we obtain a problem that is, in general, at least as difficult as the case in which a customer’s demand may be split, and which contains a more substantial additional combinatorial component.

Early exact algorithms for CFLP focus on branch-and-bound approaches. Many efforts utilized linear relaxations of CFLP to obtain upper bounds (see Sa [87] and Akinc and Kumawala [8]). Davis and Ray [25] improved upon this work by generating upper bounds from relaxations with the additional constraints

$$y_i \geq x_{ij} \quad i \in I; \quad j \in J.$$  \hspace{1cm} (2–6)

Cormueojols et al. [23] formally showed that bounds obtained using this alternative relaxation are stronger than those obtained from the standard LP relaxation of CFLP.
A separate body of research utilized Lagrangian relaxation to obtain upper bounds. Geoffrion and McBride [42], Nauss [71] and Cornuejols et al. [23] all studied Lagrangian relaxations in which single-source constraints (2–3) are relaxed. Van Roy [86] and Cornuejols et al. [23] considered the alternative Lagrangian relaxation that relaxes the capacity constraints (2–2). Interestingly, Cornuejojols et al. showed that the bound obtained by relaxing the capacity constraints is stronger than that obtained by relaxing the assignment constraints. Numerous additional methods have been proposed to solve CFLP to optimality. These include Geoffrion and Graves’ [41] implementation of Benders decomposition and Erlenkotter’s [32] study of a dual ascent method. More recently, Homberg et al. [49] proposed an exact methodology that successfully combines Lagrangian heuristics with a repeated matching algorithm to produce high quality solutions while using Lagrangian relaxations in the bounding procedure at each node. Lastly, Neebe and Rao [72] considered a branch-and-price approach which is common for assignment-based problems and is discussed in greater detail in Section 2.2. They note the difficulty of solving problems with large fixed-charges using this approach, which is a challenge that we specifically confront in Chapter 4.

Numerous heuristics for the CFLP exist. Delmaire et al. [26] considered a wide assortment of heuristic approaches, including evolutionary algorithms, tabu search (TS), simulated annealing, and a greedy randomized adaptive search procedure (GRASP). Later, Delmaire et al. [27] improved on the promising TS and GRASP heuristics and proposed different hybridization schemes that combined these two procedures, yielding quality results that require only a small amount of time. An even greater number of Lagrangian-based heuristics have been proposed for the CFLP (e.g., Barcelo and Casanovas [15], Hindi and Pienkosz [48], Klincewicz and Luss [53], and Beasley [17]). These separate efforts consider different relaxation alternatives for bounding purposes and offer unique techniques for generating feasible solutions. Of specific relevance to the methodology proposed in this work, Barcelo and Casanova [15] proposed a multi-stage
procedure that uses dual information from the linear relaxation of the CFLP to select a set of open plants before proceeding to a penalty-based reassignment procedure that, in effect, solves a specialized instance of the generalized assignment problem. As will be evident throughout Chapter 6, separating the resource procurement and demand assignment decisions into individual phases can lead to a successful heuristic framework.

More recently, Ahuja et al. [6] proposed a search heuristic which explored a very large solution space made possible by the consideration of multiple neighborhoods. Two of these neighborhoods are represented in the form of a graph and subsequently solved via a Very Large-Scale Neighborhood (VLSN) search procedure. VLSN is a search heuristic procedure shown to be extremely successful on problems with an assignment structure. For example, the Quadratic Assignment Problem (Ahuja et. al [4]), the Fleet-Assignment Problem (Ahuja et. al [2]) and the Vehicle Routing Problem (Ergun [31]) have all been solved via VLSN over the last decade. For CFLP, Ahuja et al. [6] showed that their approach solved 63 out of 71 tested instances to optimality in less than one-minute (using CPLEX to certify optimality). These tests included problems with up to 30 facilities and 200 customers. The success of this work motivates the heuristic proposed in Chapter 6. Therefore, in that chapter a more specific description of Ahuja et al.’s [6] approach is provided.

2.2 Generalized Assignment Problem

In Chapters 4–5 we propose exact and heuristic approaches for an extension of the Generalized Assignment Problem (GAP). The GAP is a special case of CFLP, in which facility procurement is not considered. That is, customer demand is fulfilled by a fixed set of capacitated resources (facilities). The GAP can be represented as

\[
\text{maximize } \sum_{i \in I} \sum_{j \in J} p_{ij} x_{ij}
\] (2–7)
subject to \[ \sum_{j \in J} a_{ij}x_{ij} \leq b_i \quad i \in I \] (2-8)

\[ \sum_{i \in I} x_{ij} = 1 \quad j \in J \] (2-9)

\[ x_{ij} \in \{0, 1\} \quad i \in I; j \in J. \] (2-10)

with parameters and variables defined in the same manner as the CFLP. Of course, the GAP is \( \mathcal{NP} \)-Hard, as shown by Fisher et al. [35]. Furthermore, the feasibility problem associated with the GAP is \( \mathcal{NP} \)-Complete, (see Martello and Toth [64]). The problem was originally studied by Ross and Soland [84], who proposed an a branch-and-bound algorithm to solve the problem to optimality. In their work, assignment constraints (2-8) are deleted and the remaining assignment problem is solved to obtain a valid upper bound. Then, a secondary penalty problem is solved to correct violated capacity restrictions. Since then, a large number of additional branch-and-bound approaches for the GAP have been proposed. These works are differentiated by the varying approaches used to bound the solution. Fisher [34] considered the strength of bounds obtained by solving (i) the Lagrangian relaxation formed by relaxing capacity constraints (2-8) (ii) the Lagrangian relaxation obtained by relaxing assignment constraints (2-9) or (iii) solving the LP relaxation formed by relaxing binary constraints (2-10). This work discusses interesting trade-offs between solving computationally difficult relaxations that provided sharper bounds, as shown to be the case with the relaxation given by (ii), versus weaker bounds obtained in less time. While this dissertation does not utilize the technique of Lagrangian relaxation, trade-offs such as these are highly relevant to other decomposition approaches that we consider.

In addition to the well-studied branch-and-bound procedure, a number of decomposition based approaches have been proposed for the GAP. Building on the Lagrangian relaxation efforts discussed previously, Jörnsten and Näsberg [52] proposed a Lagrangian decomposition
methodology that combined the two relaxations formed by relaxing either the assignment (2–9) or capacity constraints (2–8). They showed that the bound obtained by the resulting relaxation solution is at least as strong as either of the bounds obtained by the individual Lagrangian relaxation alternatives. While their testing is limited to only 10 instances, results suggested that the approach is an effective alternative to the traditional Lagrangian relaxations of the GAP.

In each of the problems studied in this dissertation, an equivalent partition-based representation is proposed with a subsequent solution methodology. Therefore, of particular relevance to the work in the dissertation is Salvesbergh’s [88] branch-and-price algorithm for the GAP. In this approach, the GAP is represented as a partition of the set of customers, \( \mathcal{J} \), into \( |\mathcal{I}| \) disjoint and possibly empty subsets, each of which is assigned to exactly one facility. That is, the formulation GAP-SP equivalently represents the GAP.

\[
\text{maximize } \sum_{i \in \mathcal{I}} \sum_{d=1}^{D_i} (p_{ij} x_{ij}^d) \lambda_i^d
\]

subject to

\[
\sum_{i \in \mathcal{I}} \sum_{d=1}^{D_i} x_{ij}^d \lambda_i^d = 1 \quad j \in \mathcal{J} \tag{2–11}
\]

\[
\sum_{d=1}^{D_i} \lambda_i^d = 1 \quad i \in \mathcal{I} \tag{2–12}
\]

\[
\lambda_i^d \in \{0, 1\} \quad d = 1, \ldots, D_i; \ i \in \mathcal{I},
\]

where \( x_i^d = (x_{i1}^d, \ldots, x_{i|\mathcal{J}|}^d) \) is a binary vector representing the \( d \text{th} \) subset of customers that can be assigned to facility \( i \), and \( D_i \) is the total number of subsets of customers that can be assigned to facility \( i \). The variable \( \lambda_i^d \) takes the value of one if the \( d \text{th} \) column associated with facility \( i \) chosen, and zero otherwise. In general, the number of variables in (GAP-SP) is exponentially large in the dimension of the underlying assignment problem. The branch-and-price approach therefore solves the LP-relaxation of (GAP-SP) by a column generation procedure, where the columns are added iteratively as needed, and
solves (GAP-SP) itself by branch-and-bound. The so-called *pricing problem* solved to identify attractive columns is the well-studied 0-1 knapsack problem, discussed in detail in Section 2.3. Salvesbergh [88] showed that the LP-relaxation of GAP-SP provides a bound at least as tight as that obtained by solving the LP-relaxation of GAP, since the feasible space of LP(GAP-SP) is limited to convex combinations of solutions to a 0-1 knapsack problem. Importantly, the work demonstrated that the branch-and-price approach is particularly successful when the ratio of customers to facilities is small (i.e. no more than 5). This phenomenon is the combined result of (i) the LP relaxation of the 0-1 knapsack problem being weaker when this ratio is small, thus solving the knapsack problems to optimality yields stronger bounds and (ii) the fact that number of feasible 0-1 knapsack solutions becomes quite large as the ratio of customers to facilities increases, thus the column generation procedure becomes highly computationally intensive as the number of customers per facility increases.

Even with the advances of exact algorithms for the GAP, it remains computationally impractical to solve very large instances. For this reason, a great deal of the literature is devoted to meta-heuristics for the GAP. A large number are mentioned in Romero Romales and Romeijn [83] summary of research pursued for the GAP. Notable amongst these are tabu search (Yagiura et. al [101]), genetic algorithms (Wilson [100]) and simulated annealing algorithms (Osman [73]). Other successes were documented in Amini and Racer’s [10] improved implementation of the variable depth search heuristic that benefits from the greedy heuristics proposed by Martello and Toth [64]. Cattrysse et al. [20] proposed a heuristic which solves the linear relaxation of the set-partitioning representation of the GAP (i.e. GAP-SP) and searches amongst the columns generated to obtain a feasible solution. The results of a computational study emperically suggest that solutions obtained in this manner are often within less than 1% of optimality.

Of particular significance to Chapter 5 is the class of heuristics proposed by Martello and Toth [64] and Romeijn and Romero Morales [79]. A weight function, $f(i, j)$, is defined
to measure the pseudo-profit of assigning customer $j$ ($j \in J$) to facility $i$ ($i \in I$). This function is used to determine the order in which to assign the customers and the facility to which each customer should be assigned. In Martello and Toth [64], weight functions are represented by either (i) the fixed profit (cost) $p_{ij}$ ($i \in I; j \in J$) (ii) the amount of resource $i$ consumed by customer $j$, $a_{ij}$ ($i \in I; j \in J$) or (iii) the ratio $\frac{a_{ij}}{b_i}$. Romeijn and Morales [79] proposed a weight function that seeks to assign a customer to a facility with maximum profit and minimal capacity consumption. To accomplish this they chose

$$f_\lambda(i,j) = p_{ij} - \lambda_i a_{ij}$$

for some vector $\lambda$. They showed that if $\lambda_i$ ($i \in I$) is taken to be the optimal dual values associated with constraints (2–8) then their greedy algorithm is optimal with probability one as the number of customers goes to infinity under a very general stochastic model. It is this work that motivates the greedy heuristic developed in Chapter 5 for which we also seek asymptotic performance guarantees.

### 2.3 0-1 Knapsack Problem

Throughout this dissertation numerous we study variants of 0-1 knapsack problems. While a large segment of our work considers 0-1 knapsack problems with non-linear profit functions, each of our solution approaches relies heavily on the properties of the linear 0-1 knapsack problem and the subsequent algorithms used to solve it. This problem, which we refer to as the KP-01, is presented as

$$\text{maximize } \sum_{j \in J} p_j x_j$$

subject to

$$\sum_{j \in J} a_{j} x_{j} \leq b$$

$$x_{ij} \in \{0, 1\}$$

$$j \in J.$$
This optimization problem requires the most profitable subset of customers to be chosen without violating the capacity constraint (2–14). Martello et al. [62] provided an excellent overview of exact algorithms proposed for KP-01. As stated in this survey, a majority of exact algorithms are based on either branch-and-bound or dynamic programing approaches. Not surprisingly, the variations in the branch-and-bound approaches consider varying procedures for obtaining good upper bounds to KP-01. The original bound on KP-01 was established in Dantzig [24]. The bound is determined by solving the linear relaxation of KP-01 in which the binary restrictions (2–15) are relaxed (say LP(KP-01)). LP(KP-01) is solved by sorting items \( j \in J \) in non-increasing order of \( \frac{p_j}{a_j} \) and including items in the solution to LP(KP-01) until either (i) all capacity \( b \) is consumed or (ii) all customers \( j \in J \), for which \( p_j > 0 \), are included in the solution to LP(KP-01). Importantly, the optimal solution to this relaxation contains at most one item which violates the binary restrictions (2–15). Thus, a heuristic approach to solving KP-01 is simply to remove the fractional customer in the optimal solution to LP(KP-01). Dantzig’s bound was later improved by Martello and Toth in [63] and then again in [67]. In the latter work, Martello and Toth [67] showed that stronger bounds can be obtained by adding maximum cardinality constraints to KP-01 and solving the corresponding relaxation via Lagrangian techniques. It is important to note that in each of the successful branch-and-bound implementations for the GAP (see also Horowitz and Sahni [50] and Nauss [70]), a depth-first enumeration scheme was chosen.

As mentioned previously, the other common approach used to solve KP-01 to optimality is dynamic programming. Pisinger [75] proposed a dynamic programming approach that relies on bounds determined by LP(KP-01). Their approach was successful in limiting the enumeration to be considered, thus yielding good lower bounds in a reasonable amount of time. To improve upon this approach, Martello and Toth [65] developed a hybrid procedure which combined the expanding-core algorithm proposed
in Pisinger [75] with the improved bounds obtained in [67]. This approach was shown to solve almost all test instances (with up to 100,000 variables) in less than .5 seconds.

## 2.4 Flexible Demand Assignment Problems

Each of the problems studied in this dissertation can be presented as generalizations of the GAP. Specifically, each problem generalizes the notion of fixed demand consumption. We mathematically introduce the concept of flexible demand in Chapter 3. This section discusses separate works in the literature that have considered the concept of variable demand fulfilled between customer-specified limits. The most relevant work in this area was offered by Balakrishnan and Geunes [13] who considered a production planning problem with flexible product specification. Their model was motivated by the steel industry, in which customers will accept steel plates cut within specified dimensions. They formulate this problem as the Flexible Demand Assignment problem (FDA), given by

\[
\begin{align*}
\text{maximize} & \quad \sum_{i \in I} \sum_{j \in J} (r_j + \chi) v_{ij} - \sum_{i \in I} (f_i + \chi b_i) y_i \\
\text{subject to} & \quad \sum_{j \in J} v_{ij} \leq b_i y_i \quad i \in I \\
& \quad \sum_{i \in I} x_{ij} = 1 \quad j \in J \\
& \quad v_{ij} \geq \ell_j x_{ij} \quad i \in I; j \in J \\
& \quad v_{ij} \leq u_j x_{ij} \quad i \in I; j \in J \\
& \quad x_{ij} \in \{0, 1\} \quad i \in I; j \in J \\
& \quad y_i \in \{0, 1\} \quad i \in I
\end{align*}
\]

where customer \( j \)'s demand level \( (j \in J) \) \( (v_{ij} \ (i \in I; j \in J)) \) must be set to a value within \([\ell_j, u_j]\). Furthermore, a revenue of \( r_j \) is accrued per unit of capacity consumption. It should be noted that the revenue accrued as a function of the level at which the demand
is satisfied is independent of the facility used to satisfy this demand. Similarly, the bounds on the demand fulfillment level are independent of the facility to which the customer is assigned. If resource $i$ is used to fulfill any customer demand a fixed cost, $f_i$ is incurred. Unused capacity of procured resource $i$ is recycled at a cost of $\chi$ per unit. In addition to the recognizable capacity and assignment constraints (2–16) and (2–17), constraints (2–18) and (2–19) ensure that demand is fulfilled between the appropriate bounds. Balakrishnan and Geunes [13] proposed multiple classes of strong valid inequalities for FDA, as well as a Lagrangian-based upper bounding procedure. Lagrangian-based, bin-packing and linear program rounding heuristics are presented. There solution procedure obtained an initial lower bound heuristically and utilized tightened relaxations to obtain an initial upper bound. These bounds are provided to CPLEX’s standard branch-and-bound procedure. A thorough computational study showed that this composite approach successfully solves both real and random problem instances up to 12 resources and 60 customers to optimality (or to within a small gap).

Beyond the FDA and the work contained in this dissertation, the most related work to flexible demand found in the literature is that done on the Multi-level Generalized Assignment Problem [57]. In this extension of the GAP, each customer is assigned to a single resource at one of a fixed number of levels. However, the model does not consider a continuous range of levels in which a customer’s demand may be fulfilled, as allowed in FDA and the problems considered in this dissertation.
Problems that require the allocation of limited resources to demands for those resources arise in nearly all contexts. In industrial contexts, the costs associated with acquiring relevant resources can often be quantified, as can the costs associated with using these resources to satisfy corresponding demands. In such cases, optimization models serve as a powerful tool for determining the best mix of resource acquisition and allocation of resources to demands. The Capacitated Facility Location Problem (CFLP) provides an example of a well-known optimization model that has been successfully utilized to determine an optimal subset of facilities (from among a set of candidate locations) as well as the allocation of the output of these facilities to a set of (known) customer demands. This problem assumes that each candidate facility has an associated fixed operating cost, a known capacity limit on output, and a cost for satisfying a customer’s demand that is proportional to the amount of the customer’s demand satisfied from the facility. Beyond the facility location context, this model finds application in a wide variety of settings in which individual capacitated resources must be acquired to satisfy demands for resource output. In this chapter, we introduce a problem that combines key elements of several well-studied variants of this problem into a new class of generalized capacitated facility location problems that we will refer to as the *Capacitated Facility Location Problem with Single-Source Constraints and Flexible Demand* (CFLFD).

As mentioned in Chapter 2, the CFLP with single-source constraints (a special case of the CFLFD) falls into the difficult class of \( \mathcal{NP} \)-Hard optimization problems, implying that it is unlikely that a polynomial-time solution method exists for solving problems in this class (unless \( \mathcal{P} = \mathcal{NP} \)). The special case of the CFLP in which there are no fixed costs associated with the acquisition of facilities constitutes a well-known problem class known as the Generalized Assignment Problem (GAP) studied by Ross and Soland [84], amongst others. This model is itself widely applicable in numerous problem contexts such as job
scheduling, location models [85], and transportation planning [92]. The classical variants of the CFLP and the GAP seek to meet known customer demand levels at minimum cost. Thus, the required level of resource consumption for each job or customer demand is assumed to be a fixed quantity. More recently, problems which consider flexible job sizes or flexible customer demand quantities have been proposed by Balakrishnan and Geunes [13]. Accounting for such flexible demands allows for scenarios common to both the steel [13] and forestry industries, for example, where customers may permit a range of delivery quantities or a range of acceptable product sizes. In such cases, the revenue received by the supplier may often increase in the delivered quantity or product size. Thus, the supplier has incentive to increase revenue by satisfying demands at the upper limits of acceptable ranges. Such a strategy, however, increases resource costs and may, therefore, not result in profit maximization.

The new class of the CFLFD problems that we introduce and analyze in this dissertation combines all of the aspects described above. In particular, this problem seeks a profit-maximizing solution based on decisions involving the procurement of capacitated resources, the assignment of customers to these resources, and the determination of corresponding demand fulfillment levels. Let $\mathcal{I}$ denote the set of facilities available for the execution of the set of customers $\mathcal{J}$. Each customer $j \in \mathcal{J}$ must be assigned to a single facility. However, each facility may only be able to process certain customers. That is, only customers in the set $\mathcal{J}_i \subseteq \mathcal{J}$ may be assigned to facility $i \in \mathcal{I}$ (or, equivalently, customer $j \in \mathcal{J}$ may only be processed by facilities in the set $\mathcal{I}_j \subseteq \mathcal{I}$, where of course $i \in \mathcal{I}_j$ if and only if $j \in \mathcal{J}_i$). If customer $j \in \mathcal{J}$ is assigned to facility $i \in \mathcal{I}_j$, a fixed profit of $p_{ij}$ is incurred and a fixed amount of capacity $a_{ij}$ is consumed. The corresponding customer demand fulfillment level must be selected from the interval $[\ell_{ij}, u_{ij}]$. An additional profit is accrued as a function of demand fulfillment level, determined by the non-decreasing function $r_{ij}$ ($i \in \mathcal{I}$; $j \in \mathcal{J}_i$). Lastly, if facility $i \in \mathcal{I}$ is used to satisfy any customer demand, a fixed procurement cost $f_i$ is incurred. The capacity of facility
$i \in I$ is denoted by $b_i$ ($i \in I$). The objective is to determine an assignment of customers to procured facilities, as well as the corresponding demand fulfillment levels, in order to maximize total profit while satisfying the capacity constraints of the facilities.

Using the preceding notation, the CFLFD can be formulated as a mixed-integer linear programming problem as follows:

\[
\text{maximize} \quad \sum_{i \in I} \sum_{j \in J} r_{ij}(v_{ij}) + \sum_{i \in I} \sum_{j \in J} p_{ij}x_{ij} - \sum_{i \in I} f_i y_i
\]

subject to

\[\sum_{j \in J} (a_{ij}x_{ij} + v_{ij}) \leq b_i y_i \quad i \in I \quad (3-1)\]
\[\sum_{i \in I} x_{ij} = 1 \quad j \in J \quad (3-2)\]
\[v_{ij} \geq \ell_{ij} x_{ij} \quad i \in I; j \in J_i \quad (3-3)\]
\[v_{ij} \leq u_{ij} x_{ij} \quad i \in I; j \in J_i \quad (3-4)\]
\[x_{ij} \in \{0, 1\} \quad i \in I; j \in J_i \quad (3-5)\]
\[y_i \in \{0, 1\} \quad i \in I. \quad (3-6)\]

Constraints (3–1) ensure that customer demand is satisfied solely by procured facilities and that the levels at which demand is satisfied satisfies each facility’s resource availability. Constraints (3–2) require the assignment of each customer to a single facility. In addition, (3–3) and (3–4) ensure that if customer $j$ is assigned to facility $i$ ($i \in I, j \in J_i$), its demand is fulfilled at a level within its respective bounds. Lastly, (3–5) and (3–6) enforce binary restrictions on the assignment variables $x_{ij}$ ($i \in I; j \in J_i$) and facility procurement variables $y_i$ ($i \in I$).

Observe that the total revenue received when customer $j$ is assigned to facility $i$ at a level of $v_{ij} \in [\ell_{ij}, u_{ij}]$ is equal to $p_{ij} + r_{ij}v_{ij}$, which corresponds to a per-unit associated price of $\frac{p_{ij}}{v_{ij}} + r_{ij}$. The model thus allows for a form of quantity discounts that may provide an incentive to customers to accept a flexible range of demand fulfillment levels. Similarly,
the total quantity of the resource that is consumed when customer \( j \) is assigned to facility \( i \) at a level of \( v_{ij} \in [\ell_{ij}, u_{ij}] \) is equal to \( a_{ij} + v_{ij} \), which corresponds to a per-unit associated resource consumption of \( \frac{a_{ij}}{v_{ij}} + 1 \). Our model can thus account for the presence of, for example, fixed customer setup times or other loss of resources at the start of a production run.

**No facility procurement costs.** In Chapters 4 and 5 we study a special case of the CFLFD in which there are no fixed facility procurement costs, i.e., \( f_i = 0 \) for all \( i \in \mathcal{I} \), so that without loss of optimality we can set \( y_i = 1 \) for all \( i \in \mathcal{I} \). We refer to this problem as the *Generalized Assignment Problem with Flexible Demand*.

\[
\max \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} p_{ij} x_{ij} + \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} r_{ij}(v_{ij}) \tag{3–7}
\]

subject to \( \sum_{j \in \mathcal{J}} a_{ij} x_{ij} + \sum_{j \in \mathcal{J}} v_{ij} \leq b_i \quad i \in \mathcal{I} \) \hspace{1cm} (3–8)

\[
\sum_{i \in \mathcal{I}} x_{ij} = 1 \quad j \in \mathcal{J}_i
\]

\[
v_{ij} \geq \ell_{ij} x_{ij} \quad i \in \mathcal{I}; j \in \mathcal{J}_i
\]

\[
v_{ij} \leq u_{ij} x_{ij} \quad i \in \mathcal{I}; j \in \mathcal{J}_i
\]

\[
x_{ij} \in \{0, 1\} \quad i \in \mathcal{I}; j \in \mathcal{J}_i.
\]

Notice that the facility procurement variables \( y_i \) \((i \in \mathcal{I})\) have been omitted from (3–7) and (3–8).

**Dynamic problems.** It is noteworthy that the CFLFD encompasses manufacturing problems with a temporal element. Specifically, consider a multi-period flexible demand assignment problem in which customers have a due date, but can be executed prior to that at the expense of a holding cost. This problem can be formulated as an instance of the CFLFD where the facilities are time-expanded; i.e., a facility \((i, t)\) represents a
single resource \((i \in I)\) available in time period \(t (t = 1, \ldots, T)\), where \(T\) is the length of the planning horizon. Clearly, customers can only be assigned to facilities representing resources that are available on or before their due date; i.e., the set \(J_{(i,t)}\) contains only customers that do not have a due date before time period \(t\).

In the following chapter we propose an exact solution approach for the CFLFD and the GAPFD. Then, in Chapters 5 and 6 we develop large-scale heuristic approaches to the CFLFD and the GAPFD.
CHAPTER 4
EXACT ALGORITHM FOR CFLFD AND GAPFD

The class of CFLFD problems introduced in Chapter 3 seeks a profit-maximizing solution based on decisions involving the procurement of capacitated resources, the assignment of customers to these resources, and the determination of corresponding customer demand fulfillment levels. In this chapter, we focus on developing an exact approach for solving this problem. Over the past decade, many nonlinear assignment problems arising in, for example, supply chain optimization, have been reformulated as set-partitioning problems, leading to branch-and-price solution approaches to such problems. Barnhart et al. [16] provide a thorough discussion of how branch-and-price can be applied to solve large integer programming models. Applications include, for example, the Generalized Assignment Problem (Savelsbergh [88]), the Fixed-Charge Assigning Users to Sources Problem (Neebe and Rao [72]), the Multi-Period Single-Sourcing Problem (Freling et al. [37]), the Continuous-time Single-Sourcing Problem (Huang et al. [51]), joint location-inventory models (Shen et al. [89]), and warehouse-retailer network design problems with joint replenishment costs (Teo and Shu [90], Romeijn et al. [82]). It is therefore not surprising that this approach can be effectively applied to the CFLFD problem class as well. However, as with any branch-and-price algorithm, a great deal of consideration must be given to the so-called pricing subproblem that arises as well as to the branching strategy used. In the CFLFD problem, the pricing subproblem takes an interesting form, resulting in a generalization of the Knapsack Problem with Flexible Items [13] or the Knapsack Problem with Variable Item Sizes [81]. We provide an efficient approach for solving this class of problems under either convex or concave revenue functions, which leads to an effective branch-and-price approach for the CFLFD problem.

This chapter is organized as follows. Section 4.1 reformulates the CFLFD as a set-partitioning problem and introduce the pricing subproblem. Section 4.2 proposes methodologies for solving a class of generalized knapsack problems which includes the
relevant pricing subproblem. Section 4.3 details the implementation of our branch-and-price algorithm. In Section 4.4 we perform an extensive computational study of the proposed procedure to solve both the CFLFD problem and the GAPFD. We include a broad collection of experiments with revenue functions that model a variety of common pricing conditions, and show that our branch-and-price algorithm significantly outperforms a state-of-the-art commercial mixed-integer nonlinear programming solver. Finally, Section 4.5 provides some concluding remarks and offers directions for future research.

4.1 Alternative Representation of the CFLFD

4.1.1 Set-Partitioning Formulation

We can equivalently view the CFLFD as a problem of partitioning the set of customers, \( J \), into \(|I|\) disjoint and possibly empty subsets, each of which is assigned to exactly one facility. More formally, we can write the set-partitioning formulation of the CFLFD as

\[
\text{maximize} \quad \sum_{i \in I} \sum_{d=1}^{D_i} \alpha_i(x^d_i) \lambda^d_i
\]

subject to

\[
\sum_{i \in I} \sum_{d=1}^{D_i} x^d_{ij} \lambda^d_i = 1 \quad j \in J \tag{4-1}
\]

\[
\sum_{d=1}^{D_i} \lambda^d_i = 1 \quad i \in I \tag{4-2}
\]

\[
\lambda^d_i \in \{0, 1\} \quad d = 1, \ldots, D_i; \ i \in I,
\]

where \( x^d_i = (x^d_{i1}, \ldots, x^d_{i|J_i|}) \) is a binary vector representing the \( d^{th} \) subset of customers that can be assigned to facility \( i \), and \( D_i \) is the total number of subsets of customers that can be assigned to facility \( i \). Furthermore, \( \alpha_i \) is a function that determines the revenue obtained by facility \( i \) when subset \( x^d_i \) is assigned to it. In particular, \( \alpha_i(x^d_i) \) is the optimal value of the following optimization problem, in which the sizes of the assigned customers are chosen to maximize the revenue of the facility:
\[
\begin{align*}
\sum_{j \in J_i} p_{ij} x_{ij}^d + & \text{ maximize } \sum_{j \in J_i} r_{ij}(v_j) - f_i y_i \\
\text{subject to } (CC_i^d) & \\
\sum_{j \in J_i} v_j & \leq b_i y_i - \sum_{j \in J_i} a_{ij} x_{ij}^d \\
v_j & \in [\ell_{ij} x_{ij}^d, u_{ij} x_{ij}^d] \quad j \in J_i \\
y_i & \in \{0, 1\}.
\end{align*}
\]

Since it is easy to see that \(\alpha_i(0) = 0\) we can, without loss of optimality, relax constraint (4–2) to

\[
\sum_{d=1}^{D_i} \lambda_i^d \leq 1 \quad i \in I. \tag{4–2'}
\]

We choose the latter set of constraints for convenience, since this immediately implies that the associated dual variables are nonnegative.

Generally, the number of variables (columns) in (SP) is exponentially large in the dimension of the underlying assignment problem. The branch-and-price approach therefore solves the LP-relaxation of (SP) by a column generation procedure, where the columns are added iteratively as needed, and solves (SP) itself by branch-and-bound. In the column generation procedure, the so-called \textit{pricing problem} determines whether the solution to the LP-relaxation of a restricted version of (SP) (in which only a subset of the columns is considered, say LP(RSP)) is indeed optimal or, otherwise, identifies one or more columns that price out and are therefore added to the restricted problem. Note that it is sufficient to either identify a feasible solution to the pricing problem that prices out \textit{or} show that the optimal solution to the pricing problem does not price out. That is, it is not strictly necessary to solve the pricing problem to optimality at each iteration of the column generation method. In the following section we formally present the pricing problem and in the subsequent section we develop approaches to solve this pricing problem under various forms of the revenue functions \(r_{ij} (i \in I; j \in J_i)\).
4.1.2 Pricing Problem

The following optimization problem describes the pricing problem associated with facility $i \in \mathcal{I}$:

$$\begin{align*}
\text{maximize} & \quad \sum_{j \in \mathcal{J}_i} (p_{ij} - \pi_j) x_j + \sum_{j \in \mathcal{J}_i} r_{ij}(v_j) - (\gamma_i + f_i) y \\
\text{subject to} & \quad \sum_{j \in \mathcal{J}_i} (a_{ij} x_j + v_j) \leq b_i y \\
& \quad v_j \in [\ell_{ij} x_j, u_{ij} x_j] \quad j \in \mathcal{J}_i \tag{4-3} \\
& \quad x_j \in \{0, 1\} \quad j \in \mathcal{J}_i \\
& \quad y \in \{0, 1\},
\end{align*}$$

where $\pi_j (j \in \mathcal{J})$ and $\gamma_i (\in \mathcal{I})$ are the optimal dual variables associated with the assignment constraints (4-1) and the column selection constraints (4-2) in LP(RSP).

Observe that when $y = 0$ the optimal solution value of (PP$_i$) is trivially seen to be equal to 0, so that we can limit ourselves to solving the problem under the assumption that facility $i$ is procured (i.e., $y = 1$). We then simply replace the optimal value to this restricted problem by zero if it is negative. To simplify the development of our solution methods, in Section 4.2 we study an equivalent formulation of the restriction of (PP$_i$) to $y = 1$. In this reformulation, we replace the demand fulfillment level variables, $v_j$, with decision variables $w_j = v_j + a_{ij} x_j (j \in \mathcal{J}_i)$, which represent the total amount of resource (i.e., both fixed and variable) consumed by customer $j$. The alternative formulation, (PP'$_i$), is written as

$$\begin{align*}
\text{maximize} & \quad \sum_{j \in \mathcal{J}_i} (p_{ij} - \pi_j) x_j + \sum_{j \in \mathcal{J}_i} r_{ij}(w_j - a_{ij} x_j) - (\gamma_i + f_i) \\
\text{subject to} & \quad \sum_{j \in \mathcal{J}_i} w_j \leq b_i \tag{4-4}
\end{align*}$$
\[ w_j \in [\ell'_{ij}x_j, u'_{ij}x_j] \quad j \in \mathcal{J}_i \quad (4-5) \]
\[ x_j \in \{0, 1\} \quad j \in \mathcal{J}_i, \quad (4-6) \]

where \( \ell'_{ij} = \ell_{ij} + a_{ij} \) and \( u'_{ij} = u_{ij} + a_{ij} \) \((i \in \mathcal{I}; j \in \mathcal{J}_i)\). It is easy to see that (4–5) assures that (4–3) remains satisfied.

This problem is a generalization of the Knapsack Problem with Expandable Items that allows for nonlinear revenue functions. Balakrishnan and Geunes [13] propose a dynamic programming algorithm for this problem for the case where the revenue functions are linear and the problem data is integer. This approach has a running time that is pseudo-polynomial in the problem inputs, which tends to make it time-consuming in practice. Additionally, their work does not consider non-linear revenue functions. Therefore, in the next section we consider approaches for solving our pricing problem under different classes of revenue functions. We develop a heuristic approach as well as a customized branch-and-bound approach to solve the problem to optimality.

### 4.2 Knapsack Problem with Expandable Items

In this section we focus on efficient methodologies to solve the class of problems given by

\[
\text{maximize} \sum_{j \in \tilde{\mathcal{J}}} p_j x_j + \sum_{j \in \mathcal{J}} r_j (w_j - a_j x_j)
\]

subject to

\[
\sum_{j \in \mathcal{J}} w_j \leq b \quad (4-7)
\]
\[ w_j \in [\ell'_{ij}x_j, u'_{ij}x_j] \quad j \in \tilde{\mathcal{J}} \quad (4-8) \]
\[ x_j \in \{0, 1\} \quad j \in \tilde{\mathcal{J}}, \quad (4-9) \]

where \( \tilde{\mathcal{J}} \) is the set of customers to be considered. We can represent (KPEI) in its most general form as

\[
\text{maximize} \sum_{j \in \mathcal{J}} \phi_j(w_j)x_j
\]
subject to \( (KPEI') \)

\[
\sum_{j \in \tilde{J}} w_j x_j \leq b \quad (4-10)
\]

\[
w_j \geq 0 \quad j \in \tilde{J} \quad (4-11)
\]

\[
x_j \in \{0, 1\} \quad j \in \tilde{J}, \quad (4-12)
\]

where the functions \( \phi_j \) are defined as

\[
\phi_j(w_j) = \begin{cases} 
0 & w_j = 0 \\
-\infty & 0 < w_j < \ell'_j \\
p_j + r_j(w_j - a_j) & \ell'_j \leq w_j \leq u'_j \\
-\infty & u'_j < w_j \leq b_i.
\end{cases}
\]

The value of \( \phi_j \) on the intervals \((0, \ell'_j)\) and \((u'_j, b)\) ensures that customers included in the solution (i.e., customers \( j \) such that \( x_j^{KPEI'} = 1 \)) will not be executed at a size outside the allowable range \([\ell'_j, u'_j]\). This consideration, along with \((4-10)\) and \((4-11)\) ensure that both \((4-7)\) and \((4-8)\) are satisfied, which ensures that \((KPEI)\) and \((KPEI')\) are equivalent.

Problem \((KPEI')\) is a Knapsack Problem with Variable Item Sizes (KP) introduced by Romeijn and Sargut [81]. They develop both a heuristic and an exact branch-and-bound approach to solve this class of optimization problems. Both methodologies rely on solving the relaxation of \((KP)\) given by

\[
\text{maximize} \quad \sum_{j \in \tilde{J}} \theta_j(w_j) \\
\text{subject to} \quad (RKP)
\]

\[
\sum_{j \in \tilde{J}} w_j \leq b \quad (4-13)
\]

\[
w_j \geq 0 \quad j \in \tilde{J},
\]
where the function \( \theta_j (j \in \tilde{J}) \) corresponds to the non-decreasing concave envelope encompassing the origin, the function \( \phi_j \) and the point \( (b, \phi_j(u'_j)) \) (which is illustrated in Figure 4-1). We will show that for the for \( \text{(KPEI') with concave, convex, or linear revenue functions, the envelopes } \theta_j (j \in \tilde{J}) \text{ can be obtained explicitly.}

Romeijn and Sargut \([81]\) propose an algorithm to solve \( \text{(RKP)} \) based on a binary search for obtaining the optimal Lagrange multiplier satisfying the KKT conditions. However, note that in the context of \( \text{(KPEI')} \), \( \text{(RKP)} \) allows solutions in which \( u'_j < u^\text{RKP}_j \leq b \ (j \in \tilde{J}) \). This, of course, corresponds to an infeasible solution. Lemma 1 ensures that an alternative feasible solution exists with an equivalent objective value.

**Lemma 1.** Suppose the optimal solution to \( \text{(RKP)} \) contains a non-empty set of customers \( \tilde{J} \) for which

\[ u'_j < u^\text{RKP}_j \leq b \ (j \in \tilde{J}). \]

There exists an alternative solution, \( \tilde{w}^\text{RKP} \), in which

\[ \tilde{w}^\text{RKP} = u'_j \ (j \in \tilde{J}) \]

and \( \tilde{z}^\text{RKP} = z^\text{RKP} \).

**Proof.** The result follows immediately from the fact that \( \theta_j = p_j + r_j(u'_j - a_j) \) over the interval \([u'_j, b]\). \( \square \)

The optimal solution of \( \text{(RKP)} \) has an attractive property which motivates an approach to solve \( \text{(KP)} \), and thus \( \text{(KPEI')} \). Specifically, by Theorem 4.2 of \([81]\), an optimal solution to \( \text{(RKP)} \) exists containing at most one customer \( j \ (j \in \tilde{J}) \) for which the function values \( \theta_j \) and \( \phi_j \) differ. In the context of \( \text{(KPEI)} \), this result can be interpreted as shown in the following lemma.

**Lemma 2.** An optimal solution to \( \text{(RKP)} \) exists in which the size of at most one customer \( j \) is such that

(i) \( w_j < l'_j \) (i.e., customer \( j \) is fractional), or
(ii) customer $j$ is fully assigned, but its net profit was overestimated (i.e., $\theta_j(w_j) > \phi(w_j)$ and $\ell'_j \leq w_j \leq u'_j$).

Proof. The result follows from Theorem 4.2 of [81] and the definition of $\theta_j$. □

This property can be used to develop an effective heuristic rounding strategy as is often done for the traditional knapsack problem. In particular, if the optimal solution to (RKP) is indeed fractional, we can simply remove that fractional customer to generate a feasible solution to the (KPEI'). Otherwise, if the solution is feasible to (KPEI'), but $\theta_j(w^\text{RKP}_j) \neq \phi_j(w^\text{RKP}_j)$ for a single customer $j$, we simply update the objective function accordingly to correct for the approximation used in the relaxation.

This property also implies that (KPEI') can be solved to optimality quite effectively using branch-and-bound even for large problem sizes (see, e.g., Martello and Toth [66]), despite the fact that it is NP-hard. The implementation of the branch-and-bound approach is as follows. A solution to (RKP) is obtained at each node of our branch-and-bound-tree, say $\tilde{w}^\text{RKP}$. We branch on the customer for which $\theta_j(\tilde{w}^\text{RKP}_j) \neq \phi_j(\tilde{w}^\text{RKP}_j)$. From Lemma 2, this corresponds to a customer that is either (i) fractionally assigned, i.e., $0 < \tilde{w}^\text{RKP}_j < \ell'_j$, or (ii) $\tilde{w}^\text{RKP}_j \geq \ell'_j$, but $\phi_j(\tilde{w}^\text{RKP}_j) < \theta_j(\tilde{w}^\text{RKP}_j)$. In case (i) we define a single branch with the constraint

$$w_j \geq \ell'_j$$

and a second branch with constraint

$$w_j = 0.$$  \hfill (4–15)

In the case of (ii) we create a branch with the constraint

$$w_j \geq \tilde{w}^\text{RKP}_j,$$  \hfill (4–16)

and another branch with the constraint

$$w_j \leq \tilde{w}^\text{RKP}_j.$$  \hfill (4–17)
Constraint (4–14) can be accommodated by first reducing $b$ by $\ell_j'$ and adding the constant $\phi(\ell_j')$ to the objective value. We then redefine the bounds of customer $j$ to be in the range $[0, u_j' - \ell_j']$ with the modified objective function component

$$\tilde{\phi}_j = \phi_j(w_j + \ell_j') - \phi_j(\ell_j') \quad 0 \leq w_j \leq (u_j' - \ell_j').$$

Constraints (4–16) and (4–17) are accounted for by simple modification of the $\phi_j$ suggested in [81]. The algorithm given in [81] can then be applied at each node of our search tree without further modification.

4.2.1 CFLFD with Specially Structured Revenue Functions

As previously discussed, the heuristic and exact branch-and-bound procedures discussed in this section solve (KPEI') with any general revenue functions $r_j$. However, the concave envelope $\theta_j$ may be difficult to characterize explicitly. Therefore, in the following sections consider important practical cases for which this can be done. In particular, we study three classes of revenue functions which model product pricing in both mass and specialized production environments. In addition to considering these functions for their real-world appeal, we show that when the revenue functions are linear or convex, (KPEI') can be more efficiently solved using an alternative procedure to solve (RKP).

Concave revenue functions. First, we consider non-decreasing concave revenue functions. This choice of function models the scenario in which marginal discounts for larger customer demand fulfillment levels are available to the customer. This scenario is applicable to manufacturers of products often purchased in bulk sizes, such as durable goods.
Now let us apply the algorithm presented in Section 4.2. In this case, the concave envelope $\theta_j (j \in \tilde{J})$ is given by

\[
\theta_j(w_j) = \begin{cases} 
\frac{\phi_j(w_j)}{\mu_j} w_j & 0 \leq w_j \leq \mu_j \\
\phi_j(w_j) & \mu_j < w_j \leq u'_j \\
\phi_j(u'_j) & u'_j < w_j \leq b,
\end{cases}
\]

where $\mu_j = \inf \left\{ s : \ell'_j \leq s \leq u'_j \text{ and } \frac{\phi_j(s - a_j)}{s} \in \partial \phi_j(s - a_j) \right\}$, $j \in \tilde{J}$

is the quantity at which the first segment of the concave envelope extending from the origin joins $\phi_j$, and $\partial \phi_j(w_j) \equiv [\partial \phi^-_j(w_j), \partial \phi^+_j(w_j)]$ is the set of subgradients of $\phi_j$ at $w_j$.

Clearly, if $\frac{\phi_j(\ell'_j)}{\ell'_j} > \partial \phi^+_j(\ell'_j)$ then $\mu_j = \ell'_j$, and if $\frac{\phi_j(u'_j)}{u'_j} < \partial \phi^-_j(u'_j)$ then $\mu_j = u'_j$. Therefore the concave envelope consists of a linear segment extending from the origin to $\mu_j$, then the true function value, $\phi_j(w_j)$ on the interval $(\mu_j, u'_j]$, and then a linear function with slope 0 on the interval $(u'_j, b]$.

In the next section, we study in greater detail two classes of revenue functions that are of equal interest in modeling specific revenue structures. In general, the method described in this section works for these revenue functions as well. However, for these special cases we will propose a more efficient algorithm.

4.2.2 Convex and Linear Revenue Functions

This section studies (KPEI) when the revenue functions are either convex or linear. We propose a more efficient algorithm to solve (RKP) that is applicable to each of these cases. The algorithm is motivated by the simplified structure of the concave envelope used in (RKP). That is, a reformulation of (RKP) can be solved in a similar manner to the continuous knapsack problem.

First, we assume convex revenue functions which are consistent with a production scenario that places increased premiums on larger customer demand fulfillment levels,
or charges an increasing amount as the amount of manufacturing time required for a
customer increases. This case is applicable to highly customized goods such as high-end
electronics or specialized automobile manufacturing.

Of course, the methods of Section 4.2 are applicable in this case as well. As is
illustrated in Figure 4-2, the concave envelope $\theta_j \ (j \in \tilde{J})$ then corresponds to a piecewise
linear function. In particular, let
$$\alpha_j = \frac{\phi_j(\ell'_j)}{\ell'_j}$$
denote the slope of the linear segment connecting the origin to the function value
evaluated at $\ell'_j$ and
$$\gamma_j = \frac{\phi_j(u'_j) - \phi_j(\ell'_j)}{u'_j - \ell'_j}$$
be the slope of the linear segment connecting the points $(\ell'_j, \phi_j(\ell'_j))$ and $(u'_j, \phi_j(u'_j))$.

Then, if $\alpha_j > \gamma_j$ we have that
$$\theta_j(w_j) = \begin{cases} 
  \alpha_j w_j & 0 \leq w_j \leq \ell'_j \\
  \alpha_j \ell'_j + \gamma_j (w_j - \ell'_j) & \ell'_j < w_j \leq u'_j \\
  \phi_j(u'_j) & u'_j < w_j \leq b 
\end{cases} \quad (4-19)$$

while if $\alpha_j \leq \gamma_j$ we have
$$\theta_j(w_j) = \begin{cases} 
  \left[ \frac{\phi_j(u'_j)}{w_j} \right] w_j & 0 \leq w_j \leq u'_j \\
  \phi_j(u'_j) & u'_j < w_j \leq b 
\end{cases} \quad (4-20)$$

Recall that by Lemma 1, without loss of generality, we need not concern ourselves
with customer demand fulfillment levels in the range $(u'_j, b)$. Therefore, we study an
alternative formulation in which we only consider customers in the range $[0, u'_j]$ . Let
$J^- = \{ j \in J : \alpha_j \leq \gamma_j \}$ and $J^+ = \{ j \in J : \alpha_j > \gamma_j \}$. We split each customer
$j \in J^+$ into two parts. The first part has demand fulfillment level $w_{j1} \in [0, \ell'_j]$ and a profit
function given by $\alpha_j w_{j1}$. The second part has demand fulfillment level $w_{j2} \in [0, u'_j - \ell'_j]$
and a revenue function given by $\gamma_j w_{j2}$. Since $\alpha_j > \gamma_j$ we will always fully utilize the entire
range corresponding to the first part of customer \( j \) before utilizing any part of the range corresponding to the second part of customer \( j \). (RKP) can then be reformulated as

\[
\text{maximize } \sum_{j \in J^-} \left( \frac{\phi_j(u'_j)}{w_j} \right) w_j + \sum_{j \in J^+} \alpha_j w_{j1} + \sum_{j \in J^+} \gamma_j w_{j2}
\]

subject to

\[
\sum_{j \in J^-} w_j + \sum_{j \in J^+} w_{j1} + \sum_{j \in J^+} w_{j2} \leq b
\]

\[
w_j \in [0, u'_j] \quad j \in J^-
\]

\[
w_{j1} \in [0, \ell'_j] \quad j \in J^+
\]

\[
w_{j2} \in [0, u'_j - \ell'_j] \quad j \in J^+.
\]

The optimal solution to this problem can be determined by simply sorting the customers \( j \in J^- \) and \( j1 \) and \( j2 \) for \( j \in J^+ \) in nonincreasing order of their coefficient in the objective function. We immediately obtain that at most one element of the optimal solution, \( w^{\text{RKP}} \), has a value that is strictly between its bounds. This implies that at most one customer is either (i) fractional or (ii) executed between its lower and upper bounds. Formally, the corresponding solution to the (RKP) is constructed by

\[
\begin{align*}
w_j^{\text{RKP}} & = w_{j1}^{\text{RKP'}} + w_{j2}^{\text{RKP'}} \quad j \in J^+ \\
w_j^{\text{RKP}} & = w_{j}^{\text{RKP'}} \quad j \in J^-.
\end{align*}
\]

This algorithm is also applicable when the revenue functions are linear. That is, a unit revenue of \( r_j \) is accrued for each unit of resource consumed; i.e., \( r_j(v_j) = r_jv_j \ (j \in \tilde{J}) \).

It is easy to see from Figure 4-3 that again the concave envelopes are piecewise linear in this case. Formally if \( p_j > 0 \) the concave envelope is given by (4–19) while if \( p_j \leq 0 \), \( \theta_j \) is given by (4–20). More interesting is the relationship identified in Theorem 1 between (RKP') and the linear relaxation of (KPEI') given by
maximize \( \sum_{j \in \tilde{J}} \bar{p}_j x_j + \sum_{j \in \tilde{J}} r_j w_j \)

subject to \( (\text{KPEI}' - R) \)

\[
\sum_{j \in J} w_j \leq b
\]

\[
w_j \in [\ell'_j x_j, u'_j x_j] \quad j \in \tilde{J}
\]

\[
x_j \in [0, 1] \quad j \in \tilde{J}
\]

where \( \bar{p}_j = p_j - r_j a_j \) \((j \in \tilde{J})\).

**Theorem 1.** The optimization problems \((\text{RKP}'\prime)\) and \((\text{KPEI}' - R)\) are equivalent when the revenue functions \(r_j\) are linear for all \(j \in \tilde{J}\).

**Proof.** See the Appendix. \(\square\)

Theorem 1 implies that the algorithm proposed in Section 4.2.2 solves the LP-relaxation of \((\text{KPEI})\). It should be noted that \((\text{KPEI}' - R)\) can be thought of as the LP-relaxation to the traditional knapsack problem, which has at most a single fractional element. This, of course, coincides with the results in Lemma 2. More interestingly, note that we need only to branch on customers which correspond to fractional assignments when the revenue functions are linear. This is best seen by revisiting Figure 4-3. Let customer \( j \) be the customer in which \( \theta_j \neq \phi_j \). If this coincides with Figure 4-3(b), then clearly \( w_j^{\text{RKP}'\prime} < \ell'_j \), which corresponds to a fractional assignment. The situation represented in Figure 4-3(a) suggests that \( w_j^{\text{RKP}'\prime} \) could take a value anywhere in the range \((0, u'_j)\). However, from Theorem 1 the solution to \((\text{RKP}'\prime)\) is equivalently the solution to \((\text{KPEI}' - R)\). From the proof of 1, \( x^{\text{KPEI}' - R} = \frac{w_j^{\text{KPEI}' - R}}{u'_j} \) for customers in which \( \theta_j \) is given by \((4-20)\) (i.e., when the case in Figure 4-3(a) holds). Thus, clearly, \( 0 < x_j^{\text{KPEI}' - R} < 1 \), so we branch on a fractional assignment in this case as well. This is significant because only branches with constraints \((4-14)\) and \((4-15)\) are necessary in the linear case, simplifying the branch-and-bound procedure described in Section 4.2.
4.3 Branch-and-Price Algorithm Implementation

Thus far we have focused primarily on how to solve (KPEI'). In this section we discuss more specific details regarding the implementation of our branch-and-price algorithm.

4.3.1 Initial Feasible Solution

Our first concern lies in providing an initial set of columns that will ensure that a feasible solution exists to the LP relaxation of the restricted set-partitioning problem, LP(RSP). If possible, we initialize RSP with feasible solutions that assign all customers to a single facility. That is, we try to initialize RSP with a column of ones (1) for each facility \( i \in \mathcal{I} \) for which the corresponding value of \( \alpha_i(1) \) is finite (i.e., for which the corresponding optimization problem defined in Section 4.3.4 is feasible). If this yields a feasible column for at least one facility \( i \in \mathcal{I} \), then we have an initial feasible solution (since, implicitly, we include a column of zeroes (0) for all facilities \( i \in \mathcal{I} \) by using the relaxed convexity constraint (4–2')).

If, as will typically be the case, it is not feasible to assign all customers to a single facility, we implement a two-phase procedure for solving (SP), where Phase 1 generates a feasible solution for LP(RSP). To this end, we include a (nonnegative) slack variable for each assignment constraint (4–1). Our Phase 1 objective is then to minimize the sum of only these slack variables. The resulting Phase 1 problem is thus given by

\[
\text{minimize } \sum_{j \in \mathcal{J}} s_j \\
\text{subject to} \\
\sum_{i \in \mathcal{I}} \sum_{d=1}^{D_i} x_{ij}^d \lambda_i^d + s_j = 1 \quad j \in \mathcal{J} \quad (4-23) \\
\sum_{d=1}^{D_i} \lambda_i^d = 1 \quad i \in \mathcal{I} \quad (4-24) \\
\lambda_i^d \in \{0, 1\} \quad d = 1, \ldots, D_i; \ i \in \mathcal{I}.\]
Here again the number of columns associated with each facility may be very large. Therefore, we solve the linear relaxation of (SP-Phase 1) using column generation. The pricing problem is similar to \((PP_i)\), except for the fact that we must account for the altered objective in (SP-Phase1). The pricing problem in Phase 1 is thus given by

\[
\text{maximize } - \sum_{j \in J} \pi_j x_j - f_i y - \delta_i
\]

subject to \((PP_i\text{-Phase1})\)

\[
\sum_{j \in J} w_j \leq b_i y \\
w_j \in [l'_{ij} x_j, u'_{ij} x_j] \quad j \in J \\
x_j \in \{0, 1\} \quad j \in J \\
y \in \{0, 1\}.
\]

This problem is a (KPEI') with revenue functions \(r_{ij} \equiv 0 \ (i \in I; \ j \in J)\). Therefore, it can be solved with the approach discussed in Section 4.2.2.

It is easy to see that if the optimal value of LP(SP-Phase1) equals 0, any optimal solution to this problem is feasible for LP(RSP); otherwise, the problem instance is infeasible. In the former case we use this feasible solution to initialize the column generation procedure for solving (SP).

### 4.3.2 Solving LP(RSP)

At any node in our branch-and-bound tree we must solve a relaxation of (SP). As previously described, this requires solving a pricing problem via the heuristic and exact methods proposed in Section 4.2. However, because our pricing problem decomposes by facility, there are \(|\mathcal{I}|\) potential pricing problems to consider. It is valid to: (i) solve them individually (in any order) and enter the first column that prices out; (ii) solve all \(|\mathcal{I}|\) problems and enter the column that prices out the highest; (iii) solve all \(|\mathcal{I}|\) problems and enter all columns that price out. In our algorithm, we choose a slightly modified version
of option (ii). At each iteration of our column generation procedure we solve all pricing problems via the heuristic method described in Section 4.2. All columns that price out favorably are added to RSP and the column generation procedure continues. We only call on our branch-and-bound procedure if all heuristic solutions indicate that none of the columns are attractive. In this case, we order the pricing problems in non-increasing order of the objective values determined by the heuristic. We continue to solve the pricing problems via branch-and-bound until either a single column prices out, or it is determined that no column prices out. This compromise enables adding multiple quality columns at each iteration via an efficient heuristic while minimizing reliance on a time-consuming branch-and-bound procedure to solve each pricing problem.

This implementation is intended to accelerate the convergence of our column generation procedure used to solve LP(RSP) at each node. However, as is common with column generation, the rate of convergence is often reduced as the optimal LP(RSP) solution is approached. To avoid this issue, we terminate our column generation procedure when our current LP(RSP) solution value is provably within $10^{-3}$ of the optimal solution to LP(RSP). This allows us to continue with our branch-and-price in a timely manner with a comparably strong upper bound with respect to that obtained if LP(RSP) was solved to optimality. Of course, to determine a valid upper bound on LP(RSP), each pricing problem must be solved to optimality. To ensure that pricing problems are not solved to optimality too often, we only update the LP(RSP) upper bound after solving $|I||J|$ pricing problems (either heuristically or exactly). This implementation choice anticipates that the number of columns required to solve LP(RSP) is a factor of both the number of facilities and the number of customers. Therefore, the frequency of updating the upper bound for LP(RSP) should decrease as either the number of customers or the number of facilities increases.

Obtaining quality feasible solutions to SP is of equal value to our branch-and-price implementation. Therefore, as a heuristic to obtain feasible solutions to SP, we solve RSP
as an integer program using columns generated in solving LP(RSP) at the root node. For a high percentage of our tests, the time required to solve this MIP is small. This implementation provided quality lower bounds early in our algorithm which proved to be beneficial in pruning our branch-and-price tree.

4.3.3 Node and Variable Selection

In our branch-and-price algorithm, we initially determine the order in which nodes should be considered by using a depth-first rule. Then, once a feasible solution to SP is obtained, we explore the tree using the well-known best-bound rule. This node selection policy is also implemented in the branch-and-bound procedure used to solve our pricing problem to optimality.

We found the branching decision to be of particular significance to our problem. It is common in the literature not to branch on the \( \lambda \) values themselves in LP(RSP) in order to preserve the structure of the pricing problem. Neebe and Rao [72] propose branching on \( x \) variables that have a value of 1 in a column associated with a fractional \( \lambda \). This branching scheme is easily accommodated in the pricing problem by generating columns adhering to any assignments fixed at previous nodes. However, as noted by Ceselli and Righini [21], intuition suggests that the impact of branching on fractional procurement variables will be greater. Note that in the context of LP(RSP), a fractional procurement variable, \( y_i \), corresponds to either \( 0 < \sum_{k=1}^{K_i} \lambda_i^k < 1 \) or \( 0 < \lambda_i^0 < 1 \), where \( \lambda_i^0 \) is a column in which no customers are assigned to facility \( i \) with cost \( \alpha_i^0 = 0 \). Though the model considered in their work makes procurement decisions with no explicit procurement cost, one of their branching strategies places a branching priority on the procurement variables. Similarly, we propose an implementation for branching on facility procurement variables fixed at 1 that preserves the structure of the pricing problem but requires no additional constraints be added to LP(RSP). That is, if \( y_i \) is fixed to 1 at a particular node, facility \( i \)'s procurement cost is simply treated as a constant in the LP(RSP) objective and the cost of columns associated with that facility are appropriately reduced by that amount.
If \( y_i \) is fixed to zero at a particular node, then all columns with customers assigned to facility \( i \) are omitted and no columns associated with facility \( i \) are generated at that node. Consistent with intuition, our testing indicated that branching priority should be given to procurement variables over assignment variables. In determining which of these \( x \)'s or \( y \)'s on which to branch, we assess the degree of fractionality of each variable in a solution to LP(RSP). The variable which is least fractional (i.e., that variable which is closest to 0 or 1, where ties are broken arbitrarily) is chosen for branching. A most-fractional variable selection implementation is also used in the pricing problem branch-and-bound algorithm.

4.3.4 Optimal Column Cost

Recall from Section 4.1 that the cost of the \( d \)th column associated with facility \( i \) (\( d = 1, \ldots, D_i; i \in I \)), is determined as a function of the optimal sizes of those customers assigned in that particular column. Of course, when the pricing problem is solved to optimality the optimal customer demand fulfillment levels are immediately available. When a column is generated using our heuristic, the size of the customers may not necessarily represent the optimal sizes for the corresponding set of customer assignments. Therefore, (CC\(_{d}^i\)) must be solved to optimality. However, note that (CC\(_{d}^i\)) can be equivalently represented by

\[
C + \text{maximize} \sum_{j \in \mathcal{J}} \bar{\theta}_j (\bar{w}_j) \bar{x}_j
\]

subject to

\[
\sum_{j \in \mathcal{J}} \bar{w}_j \bar{x}_j \leq \bar{b} \\
\bar{w}_j \geq 0 \\
\bar{x}_j \in \{0, 1\}
\]

(4.3.4)
where $\bar{J}$ is the set of customers assigned to the $d^{th}$ column associated with facility $i$, with functions $\tilde{\theta}_j$ ($j \in \bar{J}$) defined as

$$\tilde{\theta}_j(\bar{w}_j) = \begin{cases} 
0 & \bar{w}_j = 0 \\
= r_{ij}(\bar{w}_j + \ell_{ij}) - r_{ij}(\ell_{ij}) & 0 < \bar{w}_j \leq u_{ij} - \ell_{ij} \\
= r_{ij}(u_{ij}) - r_{ij}(\ell_{ij}) & u_{ij} - \ell_{ij} < \bar{w}_j \leq \bar{b},
\end{cases}$$

and with constants $C = \sum_{j \in \bar{J}} (r_{ij}(\ell_{ij}) + p_{ij}) - f_i$ and $\bar{b} = b_i - \sum_{j \in \bar{J}} (a_{ij} + \ell_{ij})$. Represented in this form, $(CC_i^d)$ simply takes the form of $(KPEI')$ and can be solved directly by the approaches discussed in section 4.2. Given an optimal solution to $(CC_i^d)$, $(\bar{x}^*, \bar{w}^*)$, the optimal customer demand fulfillment levels in terms of the original decision variables are determined by setting $w_j^* = l_j + \bar{w}_j^*$ for $j \in \bar{J}$.

### 4.4 Computational Results

In this section we discuss the performance of our branch-and-price procedure on a randomly determined set of test instances. We separately consider results for both the GAPFD and the CFLFD with varying revenue function characterizations.

#### 4.4.1 Experimental Data

In testing the non-linear representation of both the CFLFD and GAPFD we consider instances with 5 facilities, while the decreased difficulty of the linear instances allows us to consider instances with 30 facilities. In either case, instances with the number of customers equal to $|J| = 2|I|$, $3|I|$, and $5|I|$ are studied, and for the purposes of this study we assume that all customers can be assigned to all facilities. For each customer, we generate the random vectors of fixed profit parameters $P_j$ from uniform distributions on $[30, 50]$. The customer requirements $A_j$, $L_j$ and $D_j$ are generated from uniform distributions on $[10, 20]$, $[75, 125]$, and $[15, 35]$, respectively. Here, $A_j$ and $L_j$ are the random vectors of fixed capacity consumption and customer lower bounds, respectively, and $D_j$ is a random vector containing values representative of the difference between upper and lower bounds of a customer. In each of our tests we focus on instances in which the
facility capacities are identical. For GAPFD, we set $b_i = \beta |J| (i \in I)$ while for the CFLFD we generate capacities so that $b_i = \rho \beta |J| (i \in I)$, where

$$\beta = \tau \cdot \frac{E(\min_{i \in I}(A_{i1} + L_{i1}))}{|I|}. \quad (4-25)$$

The parameter $\tau$ is used in generating GAPFD instances to control the level of flexibility available when determining the size of each customer. In these tests we consider a moderate flexibility level by setting $\tau = 1.2$. The parameter $\rho > 1$, used in the generation of the CFLFD experiments, inflates the capacity of an facility to ensure that not all facilities are required in a feasible solution to the CFLFD. Without this consideration, the facility procurement decisions may be trivial. In each of the experiments considered in this section, facility capacities were generated with $\rho = 2$. Furthermore, in the case of the CFLFD, the cost of procuring an facility is directly proportional to the size of the facility itself. That is, the cost of procuring facility $i$ $(i \in I)$ is given by $F_i = b_i C_i$, where $C_i$ represents the unit cost of procurement generated from a uniform distribution on $[0.75, 1.5]$.

In our experiments, we consider the three classes of revenue functions discussed in Section 4.2. For problem instances with linear revenue functions we generate the elements of vectors $R_j$ of unit revenues using a uniform distribution on $[2, 5]$. The convex and concave revenue functions that we used in our experiments are of the form

$$r_{ij}(v_{ij}) = S_{ij}(v_{ij})^{2} \quad (4-26)$$

and

$$r_{ij}(v_{ij}) = S_{ij}\sqrt{v_{ij}}, \quad (4-27)$$

respectively. An initial value of the elements of the vector of coefficients $S_j$ for each customer is randomly generated from a uniform distribution on the interval $[0.5, 1.5]$. However, to insure that the problem instances are comparable we scale each of these coefficients so that $r_{ij}(v_{ij}) = p_{ij} + r_{ij} v_{ij}$ for $v_{ij} \in \{\ell_{ij}, u_{ij}\}$ (where $r_{ij}$ is the unit revenue
in the corresponding problem instance with linear revenue functions). The piecewise linear functions considered are obtained by approximating the concave and convex functions generated in 4–26 and 4–27. To assess the effect of varying the number of segments used to approximate the non-linear functions, we consider instances with 5, 10 and 50 segments.

Each of our instances was run until either a solution value within $10^{-3}$ of the optimal solution was obtained or a time limit of one hour was reached. Our tables present results for 10 randomly generated instances for each combination of parameter settings. All experiments were performed on a Dell power edge 2600 with two Pentium IV 3.2 Ghz processors and 6 GB of RAM. The mixed-integer programming problems as well as the relaxed master problems were solved using CPLEX 11.0. We compared the results of our approach for problems with non-linear revenue functions with those obtained by BARON 8.1.1.

In our experimentation, we sought both to assess the effectiveness of our branch-and-price approach and to gain insight on the difference in difficulty in the two non-linear cases considered. For this reason, each of our randomly generated instances was tested under both a concave and convex revenue function. Therefore, our tables associated with the results of problems with non-linear revenue functions provide combined results for each of these cases. Due to the difference in the size of instances considered in the linear case, these results are presented separately. Specifically, each table reports

(i) the number of columns generated in the entire branch-and-price algorithm,

(ii) the number of nodes considered in the branch-and-price tree

(iii) the amount of time required to solve the relaxed master problem at the root node

(iv) the total time required by the branch-and-price algorithm

(v) the total time required by the commercial solver (i.e., BARON or CPLEX), with the following additional information where appropriate:
– instances unsolved in one hour contain additional information in the superscript of the commercial solver time column;

– the first superscript indicates the relative solution error calculated by using the solvers best lower and upper bound, \( z_{\text{UB}}^S, z_{\text{LB}}^S \); i.e.,

\[
\text{error}^1 = \frac{z_{\text{UB}}^S - z_{\text{LB}}^S}{z_{\text{UB}}^S} \times 100\%;
\]

– the second superscript indicates the relative solution error calculated using the solution obtained by branch-and-price algorithm and the best solution obtained by the commercial solver, \( z_{\text{BP}}, z_{\text{LB}}^S \); i.e.,

\[
\text{error}^2 = \frac{z_{\text{BP}} - z_{\text{LB}}^S}{z_{\text{BP}}} \times 100\%.
\]

### 4.4.2 CFLFD Results

In this section we compare the performance our branch-and-price algorithm against two well-known commercial solvers on a broad set of CFLFD instances. Section 4.4.2.1 presents results for CFLFD instances with differentiable nonlinear functions solved via both the approach developed in this chapter and BARON. In Section 4.4.2.2 we perform an additional comparison for piecewise linear and linear instances solved with both branch-and-price and ILOG’s CPLEX 11.0.

#### 4.4.2.1 Nonlinear revenue functions: comparison with BARON

Tables 4-1–4-3 present results for the differentiable nonlinear (concave and convex) CFLFD instances with functions generated by (4–26) and (4–27). In each of these three tables it is evident that branch-and-price outperforms the commercial solver in every instance. Our computational study found that our choice of branching strategy, discussed in Section 4.3.3, was, in part, a contributing factor to this success. As an example, for the instances considered in Table 4-4, placing a branching priority on the procurement variables over the assignment variables reduced our average branch-and-price time by a factor of 50 for the convex instances. This is likely a direct result of the number of nodes being reduced by a factor of more than 10.
An analysis of the performance of the commercial solver indicates that BARON is unable to solve any of the concave instances to within the desired tolerance levels within one hour. When convex revenue functions are considered, BARON’s performance improves, solving the smaller set of instances shown in Table 4-1 within an average of approximately one minute. Even for the largest number of customers considered, BARON solves 7 of 10 convex instances in an average of approximately 32 minutes. Notably, our branch-and-price algorithm requires on average no more than two seconds to solve problems with $|\mathcal{J}| \leq 3|\mathcal{I}|$. Table 4-3 indicates that the branch-and-price time required grows notably when $|\mathcal{J}| = 5|\mathcal{I}|$. However, each of these instances could still be solved within the desired optimality tolerances within an hour. Our testing revealed that the extensive times required for experiments shown in Table 4-3 were not a result of the column generation procedure at the root node or the time spent solving the MIP using columns generated at the root node. Unfortunately, the time spent solving pricing problems to optimality in the subsequent nodes was the direct cause of the increased time requirements. Instances with $|\mathcal{J}| = 10|\mathcal{I}|$ customers were considered in the experimentation, but neither branch-and-price, nor BARON could solve these instances consistently within one hour. However, branch-and-price only required seconds to solve instances with 10 facilities when customer ratios were limited to $|\mathcal{J}| \leq 3|\mathcal{I}|$.

For those instances which BARON failed to solve to optimality within one hour, the quality of the best found solutions was notably poor. For smaller concave instances (i.e., $|\mathcal{J}| \leq 3|\mathcal{I}|$) the relative error between the best upper bound and incumbent solution ranges between 55% and 74%. However, for each of these instances, branch-and-price solved the problems to optimality in a matter of seconds. Even when the true optimal solution obtained by branch-and-price is used to assess the quality of the BARON solution, the average relative error only decreases to between 8% and 34%. These results were sufficient to determine that running concave experiments with $|\mathcal{J}| = 5|\mathcal{I}|$ on the commercial solver was unnecessary; therefore, the corresponding column is empty in Table
4-3. As noted previously, BARON was more successful in solving the convex instances we considered. Therefore, even for the four instances with $|J| = 5|I|$ that could not be solved in one hour, the corresponding errors are substantially reduced from those obtained in any of the concave instances. Somewhat unexpectedly, the results from the commercial solver indicate that problem instances with concave revenue functions were far more difficult than the ones with convex revenue functions. The difference between these two types of revenue functions is less pronounced when comparing the performance of the branch-and-price algorithm. Tables 4-1–4-3 indicate that the number of columns, number of nodes, and solution time in both cases were consistent in each problem set considered.

In addition to the contribution of the branching rule discussed previously, the success of our branch-and-price algorithm for nonlinear revenue functions can be attributed to the combined effectiveness of the proposed methodologies to solve our pricing problem and the tightness of the set-partitioning formulation. The latter is evident by the limited number of nodes explored in the search tree. Each of the Tables 4-1–4-3 indicates that no more than 30 nodes were considered for any single instance. Interestingly, 5 of the 10 instances presented in Table 4-1 were solved at the root node.

4.4.2.2 Piecewise linear and linear revenue functions: comparison with CPLEX

Tables 4-5–4-10 compare the performance of our branch-and-price approach for convex and concave piecewise linear functions against linearized formulations solved in CPLEX. The results shown in Tables 4-5 and 4-6 indicate that branch-and-price solves all piecewise linear convex instances with a customer-to-facility ratio less than or equal to three, regardless of the number of segments comprising the piecewise linear function, in less than one second. However, CPLEX requires at least 14 times the average computational time of branch-and-price for instances with $|J| \leq 3|I|$. For piecewise linear convex instances with 25 customers, the branch-and-price times grow notably. However, Table 4-7 shows our algorithm, on average, outperforms CPLEX for this scenario, as well. In
addition, it is clear that as the number of segments increases, CPLEX requires notably more computational time, which is a result of the additional binary variables required to model piecewise linear convex functions. To the contrary, Tables 4-5 and 4-6 suggest that the number of segments has little effect on the time required by branch-and-price. While Table 4-7 does indicate an increasing trend in average time required by branch-and-price as the number of segments increases, an analysis of individual instances suggests this trend is found only in experiments 2 and 6. Tables 4-8–4-10 show that branch-and-price is competitive with CPLEX on piecewise linear concave instances with $|\mathcal{J}| \leq 3|\mathcal{I}|$. However, CPLEX performs significantly better on concave piecewise linear instances when compared to the piecewise linear convex results. This result is expected, since piecewise linear concave revenue functions can be modeled with additional continuous decision variables and linear constraints, instead of the binary variables required for piecewise linear convex functions. Table 4-10 demonstrates that CPLEX is a better alternative as the ratio of customers to facilities increases for piecewise linear concave instances.

We also included linear instances in our computational testing. Our testing indicated that CPLEX was more successful than branch-and-price on problems with less than 30 facilities and a customer-to-facility ratio of 2, 3, or 5. However, as shown in Table 4-11, CPLEX is unable to solve any of the 10 instances with 30 facilities and 60 customers within an hour, while our branch-and-price methodology solves each of these instances in an average of less than 15 seconds. Unfortunately, our tests on larger instances (i.e., 30 facilities with 90 and 150 customers) revealed than neither CPLEX nor branch-and-price was successful within the specified time limit.

4.4.3 GAPFJ Results

In this section we present results for the GAPFJ, which is a special case of the CFLFD when facility procurement decisions are omitted from the model. For these problems, we chose to focus on (differentiable) nonlinear and purely linear revenue functions. The nonlinear results shown in Tables 4-12–4-14 indicate that the branch-and-price
algorithm performs equally as well for the GAPFJ as it did for the CFLFD. However, as indicated by Table 4-15, CPLEX outperforms our branch-and-price for linear instances of the GAPFJ. The success of CPLEX was evident for instances with customer-to-facility ratios of $2|\mathcal{I}|$ and $5|\mathcal{I}|$ as well. A comparison of Table 4-12 to Table 4-1 suggests that BARON is more successful on concave instances which omit the binary facility procurement variables. In contrast, the average time to solve convex GAPFJ instances with $|\mathcal{J}| = 2|\mathcal{I}|$ is more than that required for the CFLFD instances with the same number of facilities and customers. Note that a different set of data was used in our GAPFJ experiments, so caution should be taken in comparing results from the general CFLFD and the GAPFJ special case. However, the GAPFJ concave experiments were still clearly difficult for BARON. Only 9 of the 30 experiments considered were solved within the allotted time. In 3 of the 10 experiments shown in Table 4-14, BARON was unable to even find a feasible solution within an hour. In contrast, while only 11 of the 30 convex instances were solved by the commercial solver, the errors of the best solution obtained were much better than those obtained in the concave tests. In fact, in 12 of the 19 instances unsolved in Tables 4-13 and 4-14, the best solutions obtained were actually optimal, but BARON had yet to prove optimality when the time limit was reached.

The branch-and-price times in Tables 4-12–4-14 indicate the success of the proposed methodology on this important special case. For example, the results corresponding to $|\mathcal{J}| \leq 3|\mathcal{I}|$ show that the average branch-and-price time was only a fraction of a second. The same set of tables reveals that the number of nodes considered was minimal. In fact, in 14 of the 15 instances considered in these three tables, the problems were solved at the root node. The increase in branch-and-price time for $|\mathcal{J}| = 5|\mathcal{I}|$ is less significant than the results obtained for the CFLFD.

Testing on larger instances indicated that problems with $|\mathcal{J}| = 10|\mathcal{I}|$ customers remain solvable in under an hour. In addition, if the customer-to-facility ratio is less than three, then branch-and-price is successful in solving instances with 10 facilities in less
than 15 minutes. This suggests that the customer-to-facility ratio is the key factor in determining the size of problem instances in which branch-and-price is successful.

4.5 Conclusions

In this chapter we considered a generalization of the capacitated facility location problem that includes considerations for flexible demand. We proposed an exact branch-and-price algorithm to solve the resulting CFLFD problem based on a set-partitioning representation of our model. To solve the resulting pricing problem, we studied an interesting generalization of nonlinear knapsack problems with flexible item sizes. Motivated by a relevant relaxation with an optimal solution shown to possess attractive structural properties, we discussed how both the heuristic and exact approaches used to solve the resulting KP [81] can be applied to solve our pricing problem in its most general form. Then, for revenue structures common to real-world pricing conditions, we proposed an alternative pricing problem solution methodology, which is more efficient than that required in the most general case. Our computational study suggests that the branch-and-price approach proposed in this work consistently outperforms a popular commercial nonlinear solver in our testing of nonlinear CFLFD and GAPFJ instances.

Table 4-1. CFLFD with nonlinear revenue functions: 5 facilities, 10 customers, $\tau = 1.2$

<table>
<thead>
<tr>
<th>Exp</th>
<th>Cols</th>
<th>Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>BARON total time (sec)</th>
<th>Cols</th>
<th>Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>BARON total time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>140</td>
<td>13</td>
<td>0.2</td>
<td>0.8</td>
<td>3600$^{(57,12)}$</td>
<td>316</td>
<td>13</td>
<td>0.1</td>
<td>0.3</td>
<td>29.5</td>
</tr>
<tr>
<td>2</td>
<td>85</td>
<td>9</td>
<td>0.1</td>
<td>0.2</td>
<td>3600$^{(61,21)}$</td>
<td>83</td>
<td>9</td>
<td>0.1</td>
<td>0.1</td>
<td>46.2</td>
</tr>
<tr>
<td>3</td>
<td>198</td>
<td>33</td>
<td>0.1</td>
<td>1.8</td>
<td>3600$^{(62,17)}$</td>
<td>474</td>
<td>33</td>
<td>0.1</td>
<td>0.5</td>
<td>116.2</td>
</tr>
<tr>
<td>4</td>
<td>156</td>
<td>21</td>
<td>0.1</td>
<td>1.0</td>
<td>3600$^{(56,13)}$</td>
<td>230</td>
<td>21</td>
<td>0.1</td>
<td>0.4</td>
<td>55.6</td>
</tr>
<tr>
<td>5</td>
<td>107</td>
<td>7</td>
<td>0.1</td>
<td>0.4</td>
<td>3600$^{(57,18)}$</td>
<td>105</td>
<td>7</td>
<td>0.1</td>
<td>0.2</td>
<td>21.7</td>
</tr>
<tr>
<td>6</td>
<td>141</td>
<td>21</td>
<td>0.1</td>
<td>0.9</td>
<td>3600$^{(55,8)}$</td>
<td>140</td>
<td>21</td>
<td>0.1</td>
<td>0.3</td>
<td>70.6</td>
</tr>
<tr>
<td>7</td>
<td>118</td>
<td>13</td>
<td>0.2</td>
<td>0.6</td>
<td>3600$^{(55,13)}$</td>
<td>134</td>
<td>13</td>
<td>0.1</td>
<td>0.2</td>
<td>22.7</td>
</tr>
<tr>
<td>8</td>
<td>106</td>
<td>9</td>
<td>0.3</td>
<td>0.7</td>
<td>3600$^{(55,9)}$</td>
<td>112</td>
<td>9</td>
<td>0.1</td>
<td>0.3</td>
<td>41.3</td>
</tr>
<tr>
<td>9</td>
<td>188</td>
<td>29</td>
<td>0.1</td>
<td>1.6</td>
<td>3600$^{(60,13)}$</td>
<td>471</td>
<td>29</td>
<td>0.1</td>
<td>0.5</td>
<td>124.8</td>
</tr>
<tr>
<td>10</td>
<td>170</td>
<td>23</td>
<td>0.1</td>
<td>1.2</td>
<td>3600$^{(60,20)}$</td>
<td>271</td>
<td>21</td>
<td>0.1</td>
<td>0.4</td>
<td>99.6</td>
</tr>
<tr>
<td>Avg</td>
<td>140.9</td>
<td>17.8</td>
<td>0.1</td>
<td>0.9</td>
<td>3600$^{(58,14)}$</td>
<td>233.6</td>
<td>17.6</td>
<td>0.1</td>
<td>0.3</td>
<td>62.8</td>
</tr>
</tbody>
</table>
Figure 4-1. Illustration of $\phi_j$ and $\theta_j$ for a general revenue function

Figure 4-2. Concave envelope: convex revenue function
Figure 4-3. Concave envelope: linear revenue function

Table 4-2. CFLFD with nonlinear revenue functions: 5 facilities, 15 customers, $\tau = 1.2$

<table>
<thead>
<tr>
<th>Exp</th>
<th>Cols</th>
<th>Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>BARON time</th>
<th>Cols</th>
<th>Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>BARON time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>132</td>
<td>1</td>
<td>0.2</td>
<td>0.2</td>
<td>$3600^{(71,29)}$</td>
<td>154</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>27.9</td>
</tr>
<tr>
<td>2</td>
<td>141</td>
<td>1</td>
<td>1.1</td>
<td>1.1</td>
<td>$3600^{(72,31)}$</td>
<td>154</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>75.8</td>
</tr>
<tr>
<td>3</td>
<td>139</td>
<td>1</td>
<td>0.2</td>
<td>0.2</td>
<td>$3600^{(71,31)}$</td>
<td>156</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>96.0</td>
</tr>
<tr>
<td>4</td>
<td>348</td>
<td>19</td>
<td>0.4</td>
<td>4.4</td>
<td>$3600^{(74,34)}$</td>
<td>363</td>
<td>19</td>
<td>0.2</td>
<td>1.2</td>
<td>621.9</td>
</tr>
<tr>
<td>5</td>
<td>139</td>
<td>1</td>
<td>0.2</td>
<td>0.2</td>
<td>$3600^{(69,25)}$</td>
<td>129</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>51.9</td>
</tr>
<tr>
<td>6</td>
<td>135</td>
<td>1</td>
<td>0.3</td>
<td>0.3</td>
<td>$3600^{(68,23)}$</td>
<td>139</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>101.1</td>
</tr>
<tr>
<td>7</td>
<td>278</td>
<td>15</td>
<td>0.3</td>
<td>1.7</td>
<td>$3600^{(68,23)}$</td>
<td>313</td>
<td>15</td>
<td>0.1</td>
<td>0.7</td>
<td>232.9</td>
</tr>
<tr>
<td>8</td>
<td>224</td>
<td>9</td>
<td>0.3</td>
<td>2.2</td>
<td>$3600^{(69,28)}$</td>
<td>266</td>
<td>9</td>
<td>0.3</td>
<td>1.1</td>
<td>75.1</td>
</tr>
<tr>
<td>9</td>
<td>137</td>
<td>3</td>
<td>0.3</td>
<td>0.3</td>
<td>$3600^{(68,21)}$</td>
<td>177</td>
<td>5</td>
<td>0.1</td>
<td>0.2</td>
<td>154.5</td>
</tr>
<tr>
<td>10</td>
<td>276</td>
<td>15</td>
<td>0.5</td>
<td>5.0</td>
<td>$3600^{(70,23)}$</td>
<td>344</td>
<td>17</td>
<td>0.2</td>
<td>1.5</td>
<td>324.2</td>
</tr>
</tbody>
</table>

Avg | 194.9 | 6.6  | 0.4 | 1.6 | $3600^{(70,27)}$ | 219.5 | 7 | 0.1 | 0.5 | 176.1 |
Table 4-3. CFLFD with nonlinear revenue functions: 5 facilities, 25 customers, $\tau = 1.2$

<table>
<thead>
<tr>
<th>Exp</th>
<th>Colss</th>
<th>Nodes</th>
<th>Root</th>
<th>BP total</th>
<th>BARON</th>
<th>Colss</th>
<th>Nodes</th>
<th>Root</th>
<th>BP total</th>
<th>BARON</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>time (sec)</td>
<td>time (sec)</td>
<td>time (sec)</td>
<td></td>
<td></td>
<td>time (sec)</td>
<td>time (sec)</td>
<td>time (sec)</td>
</tr>
<tr>
<td>1</td>
<td>863</td>
<td>17</td>
<td>2.3</td>
<td>177.3</td>
<td>-</td>
<td>623</td>
<td>11</td>
<td>1.3</td>
<td>19.8</td>
<td>656.9</td>
</tr>
<tr>
<td>2</td>
<td>730</td>
<td>9</td>
<td>418.4</td>
<td>1393.9</td>
<td>-</td>
<td>672</td>
<td>9</td>
<td>17.3</td>
<td>166.2</td>
<td>600.9</td>
</tr>
<tr>
<td>3</td>
<td>1090</td>
<td>5</td>
<td>1.2</td>
<td>1128.3</td>
<td>-</td>
<td>1627</td>
<td>25</td>
<td>0.6</td>
<td>30.3</td>
<td>3600(6,2)</td>
</tr>
<tr>
<td>4</td>
<td>1299</td>
<td>23</td>
<td>4.2</td>
<td>847.6</td>
<td>-</td>
<td>1554</td>
<td>23</td>
<td>1.5</td>
<td>634.7</td>
<td>3600(4,1)</td>
</tr>
<tr>
<td>5</td>
<td>1083</td>
<td>23</td>
<td>3.1</td>
<td>183.4</td>
<td>-</td>
<td>1212</td>
<td>23</td>
<td>0.7</td>
<td>55.4</td>
<td>2931.6</td>
</tr>
<tr>
<td>6</td>
<td>1217</td>
<td>23</td>
<td>4.2</td>
<td>417.5</td>
<td>-</td>
<td>1358</td>
<td>27</td>
<td>0.7</td>
<td>87.3</td>
<td>2878.7</td>
</tr>
<tr>
<td>7</td>
<td>1080</td>
<td>19</td>
<td>2.9</td>
<td>285.3</td>
<td>-</td>
<td>1358</td>
<td>19</td>
<td>1.1</td>
<td>17.1</td>
<td>3600(4,1)</td>
</tr>
<tr>
<td>8</td>
<td>1136</td>
<td>25</td>
<td>12.4</td>
<td>586.3</td>
<td>-</td>
<td>1207</td>
<td>21</td>
<td>3.5</td>
<td>1229.8</td>
<td>2716.7</td>
</tr>
<tr>
<td>9</td>
<td>727</td>
<td>13</td>
<td>5.9</td>
<td>36.2</td>
<td>-</td>
<td>930</td>
<td>15</td>
<td>18.8</td>
<td>698.2</td>
<td>913.2</td>
</tr>
<tr>
<td>10</td>
<td>837</td>
<td>17</td>
<td>4.9</td>
<td>108.7</td>
<td>-</td>
<td>1133</td>
<td>17</td>
<td>2.4</td>
<td>335.6</td>
<td>3041.1</td>
</tr>
<tr>
<td>Averages</td>
<td>1006.2</td>
<td>19.4</td>
<td>46.0</td>
<td>516.4</td>
<td>-</td>
<td>1163.0</td>
<td>18.8</td>
<td>4.9</td>
<td>327.4</td>
<td>2088.9</td>
</tr>
</tbody>
</table>

Table 4-4. Branching rule comparison for the CFLFD with nonlinear revenue functions: 5 facilities, 10 customers, $\tau = 1.2$

<table>
<thead>
<tr>
<th>Exp</th>
<th>Nodes</th>
<th>BP total time (sec)</th>
<th>Nodes</th>
<th>BP total time (sec)</th>
<th>Nodes</th>
<th>BP total time (sec)</th>
<th>Nodes</th>
<th>BP total time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>61</td>
<td>2.8</td>
<td>13</td>
<td>0.5</td>
<td>81</td>
<td>1.8</td>
<td>13</td>
<td>0.1</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>0.2</td>
<td>9</td>
<td>0.1</td>
<td>3</td>
<td>0.2</td>
<td>9</td>
<td>0.1</td>
</tr>
<tr>
<td>3</td>
<td>291</td>
<td>7.9</td>
<td>33</td>
<td>0.9</td>
<td>321</td>
<td>6.8</td>
<td>33</td>
<td>0.1</td>
</tr>
<tr>
<td>4</td>
<td>99</td>
<td>2.5</td>
<td>21</td>
<td>0.6</td>
<td>91</td>
<td>2.6</td>
<td>21</td>
<td>0.1</td>
</tr>
<tr>
<td>5</td>
<td>11</td>
<td>0.5</td>
<td>7</td>
<td>0.3</td>
<td>14</td>
<td>0.6</td>
<td>7</td>
<td>0.1</td>
</tr>
<tr>
<td>6</td>
<td>555</td>
<td>8.2</td>
<td>21</td>
<td>0.6</td>
<td>501</td>
<td>11.9</td>
<td>21</td>
<td>0.1</td>
</tr>
<tr>
<td>7</td>
<td>69</td>
<td>1.7</td>
<td>13</td>
<td>0.3</td>
<td>85</td>
<td>2.9</td>
<td>13</td>
<td>0.1</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>0.8</td>
<td>9</td>
<td>0.5</td>
<td>9</td>
<td>1.1</td>
<td>9</td>
<td>0.1</td>
</tr>
<tr>
<td>9</td>
<td>665</td>
<td>10.9</td>
<td>29</td>
<td>0.9</td>
<td>765</td>
<td>18.7</td>
<td>29</td>
<td>0.1</td>
</tr>
<tr>
<td>10</td>
<td>109</td>
<td>2.3</td>
<td>23</td>
<td>0.8</td>
<td>107</td>
<td>3.0</td>
<td>21</td>
<td>0.1</td>
</tr>
<tr>
<td>Averages</td>
<td>187</td>
<td>3.78</td>
<td>17.8</td>
<td>0.6</td>
<td>197.7</td>
<td>4.96</td>
<td>17.6</td>
<td>0.1</td>
</tr>
</tbody>
</table>
Table 4-5. CFLFD with piecewise linear convex revenue functions: 5 facilities, 10 customers, $\tau = 1.2$

| Exp | 5 segments | | 10 segments | | 50 segments | |
|-----|------------|------------|------------|------------|------------|
|     | BP total time (sec) | CPLEX time (sec) | BP total time (sec) | CPLEX time (sec) | BP total time (sec) | CPLEX time (sec) |
| 1   | 0.5        | 1.4        | 0.4        | 0.7        | 0.4        | 2.3        |
| 2   | 0.1        | 1.1        | 0.1        | 1.6        | 0.1        | 6.1        |
| 3   | 0.9        | 4.0        | 0.8        | 4.1        | 1.1        | 10.7       |
| 4   | 0.8        | 2.5        | 0.5        | 2.4        | 0.6        | 8.1        |
| 5   | 0.3        | 1.5        | 0.2        | 0.7        | 0.3        | 3.9        |
| 6   | 0.7        | 13.6       | 0.5        | 12.9       | 0.7        | 289.5      |
| 7   | 0.5        | 4.2        | 0.4        | 3.6        | 0.5        | 15.5       |
| 8   | 0.7        | 5.4        | 0.4        | 3.6        | 0.6        | 24.9       |
| 9   | 1.1        | 47.2       | 0.8        | 20.1       | 0.9        | 241.4      |
| 10  | 0.8        | 9.9        | 0.4        | 6.0        | 0.9        | 330.0      |
| Avg | 0.6        | 9.1        | 0.4        | 5.6        | 0.6        | 93.2       |

Table 4-6. CFLFD with piecewise linear convex revenue functions: 5 facilities, 15 customers, $\tau = 1.2$

| Exp | 5 segments | | 10 segments | | 50 segments | |
|-----|------------|------------|------------|------------|------------|
|     | BP total time (sec) | CPLEX time (sec) | BP total time (sec) | CPLEX time (sec) | BP total time (sec) | CPLEX time (sec) |
| 1   | 0.2        | 0.8        | 0.1        | 0.8        | 0.3        | 2.0        |
| 2   | 0.3        | 7.5        | 0.1        | 4.7        | 0.3        | 51.2       |
| 3   | 0.1        | 1.1        | 0.1        | 2.5        | 0.2        | 3.2        |
| 4   | 1.9        | 5.7        | 1.0        | 5.4        | 2.0        | 22.7       |
| 5   | 0.1        | 1.1        | 0.1        | 0.7        | 0.2        | 2.9        |
| 6   | 0.2        | 3.3        | 0.2        | 2.0        | 0.3        | 6.9        |
| 7   | 1.0        | 14.9       | 0.8        | 9.8        | 1.2        | 49.8       |
| 8   | 1.2        | 43.3       | 0.6        | 241.4      | 1.1        | 1052.0     |
| 9   | 0.3        | 4.4        | 0.3        | 3.0        | 0.5        | 16.0       |
| 10  | 2.8        | 17.3       | 2.1        | 15.2       | 2.8        | 62.2       |
| Avg | 0.8        | 9.9        | 0.5        | 28.5       | 0.9        | 126.9      |
Table 4-7. CFLFD with piecewise linear convex revenue functions: 5 facilities, 25 customers, $\tau = 1.2$

<table>
<thead>
<tr>
<th>Exp</th>
<th>5 segments</th>
<th>10 segments</th>
<th>50 segments</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BP total</td>
<td>CPLEX</td>
<td>BP total</td>
</tr>
<tr>
<td></td>
<td>(sec)</td>
<td>(sec)</td>
<td>(sec)</td>
</tr>
<tr>
<td>1</td>
<td>43.8</td>
<td>103.9</td>
<td>26.2</td>
</tr>
<tr>
<td>2</td>
<td>24.7</td>
<td>8.9</td>
<td>131.2</td>
</tr>
<tr>
<td>3</td>
<td>26.3</td>
<td>100.4</td>
<td>76.2</td>
</tr>
<tr>
<td>4</td>
<td>307.5</td>
<td>257.8</td>
<td>977.8</td>
</tr>
<tr>
<td>5</td>
<td>54.6</td>
<td>94.8</td>
<td>158.4</td>
</tr>
<tr>
<td>6</td>
<td>177.4</td>
<td>104.3</td>
<td>261.0</td>
</tr>
<tr>
<td>7</td>
<td>31.5</td>
<td>510.5</td>
<td>19.0</td>
</tr>
<tr>
<td>8</td>
<td>71.1</td>
<td>223.7</td>
<td>88.6</td>
</tr>
<tr>
<td>9</td>
<td>42.4</td>
<td>7.8</td>
<td>42.0</td>
</tr>
<tr>
<td>10</td>
<td>69.7</td>
<td>55.3</td>
<td>26.5</td>
</tr>
<tr>
<td>Avg</td>
<td>84.9</td>
<td>146.7</td>
<td>180.7</td>
</tr>
</tbody>
</table>

Table 4-8. CFLFD with piecewise linear concave revenue functions: 5 facilities, 10 customers, $\tau = 1.2$

<table>
<thead>
<tr>
<th>Exp</th>
<th>5 segments</th>
<th>10 segments</th>
<th>50 segments</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BP total</td>
<td>CPLEX</td>
<td>BP total</td>
</tr>
<tr>
<td></td>
<td>(sec)</td>
<td>(sec)</td>
<td>(sec)</td>
</tr>
<tr>
<td>1</td>
<td>0.2</td>
<td>0.7</td>
<td>0.2</td>
</tr>
<tr>
<td>2</td>
<td>0.1</td>
<td>0.7</td>
<td>0.1</td>
</tr>
<tr>
<td>3</td>
<td>0.7</td>
<td>2.9</td>
<td>0.7</td>
</tr>
<tr>
<td>4</td>
<td>0.4</td>
<td>1.4</td>
<td>0.4</td>
</tr>
<tr>
<td>5</td>
<td>0.2</td>
<td>0.6</td>
<td>0.2</td>
</tr>
<tr>
<td>6</td>
<td>0.4</td>
<td>2.6</td>
<td>0.5</td>
</tr>
<tr>
<td>7</td>
<td>0.2</td>
<td>1.1</td>
<td>0.3</td>
</tr>
<tr>
<td>8</td>
<td>0.3</td>
<td>1.7</td>
<td>0.3</td>
</tr>
<tr>
<td>9</td>
<td>0.7</td>
<td>4.4</td>
<td>0.8</td>
</tr>
<tr>
<td>10</td>
<td>0.6</td>
<td>2.3</td>
<td>0.6</td>
</tr>
<tr>
<td>Avg</td>
<td>0.4</td>
<td>1.8</td>
<td>0.4</td>
</tr>
</tbody>
</table>
Table 4-9. CFLFD with piecewise linear concave revenue functions: 5 facilities, 15 customers, $\tau = 1.2$

<table>
<thead>
<tr>
<th>Exp</th>
<th>5 segments BP total time (sec)</th>
<th>CPLEX time (sec)</th>
<th>10 segments BP total time (sec)</th>
<th>CPLEX time (sec)</th>
<th>50 segments BP total time (sec)</th>
<th>CPLEX time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.1</td>
<td>0.5</td>
<td>0.1</td>
<td>0.6</td>
<td>0.1</td>
<td>1.1</td>
</tr>
<tr>
<td>2</td>
<td>0.2</td>
<td>0.8</td>
<td>0.2</td>
<td>0.5</td>
<td>0.6</td>
<td>1.8</td>
</tr>
<tr>
<td>3</td>
<td>0.1</td>
<td>0.6</td>
<td>0.1</td>
<td>0.6</td>
<td>0.2</td>
<td>1.3</td>
</tr>
<tr>
<td>4</td>
<td>1.0</td>
<td>1.3</td>
<td>1.5</td>
<td>2.2</td>
<td>3.5</td>
<td>2.7</td>
</tr>
<tr>
<td>5</td>
<td>0.1</td>
<td>0.6</td>
<td>0.1</td>
<td>0.4</td>
<td>0.1</td>
<td>0.8</td>
</tr>
<tr>
<td>6</td>
<td>0.1</td>
<td>1.0</td>
<td>0.1</td>
<td>1.4</td>
<td>0.3</td>
<td>2.5</td>
</tr>
<tr>
<td>7</td>
<td>0.6</td>
<td>1.4</td>
<td>0.8</td>
<td>1.4</td>
<td>1.7</td>
<td>4.5</td>
</tr>
<tr>
<td>8</td>
<td>0.7</td>
<td>0.9</td>
<td>1.2</td>
<td>0.8</td>
<td>1.2</td>
<td>1.6</td>
</tr>
<tr>
<td>9</td>
<td>0.1</td>
<td>1.2</td>
<td>0.1</td>
<td>2.0</td>
<td>0.3</td>
<td>3.4</td>
</tr>
<tr>
<td>10</td>
<td>2.0</td>
<td>4.7</td>
<td>1.7</td>
<td>4.4</td>
<td>4.5</td>
<td>10.1</td>
</tr>
<tr>
<td>Avg</td>
<td>0.5</td>
<td>1.3</td>
<td>0.6</td>
<td>1.4</td>
<td>1.2</td>
<td>3.0</td>
</tr>
</tbody>
</table>

Table 4-10. CFLFD with piecewise linear concave revenue functions: 5 facilities, 25 customers, $\tau = 1.2$

<table>
<thead>
<tr>
<th>Exp</th>
<th>5 segments BP total time (sec)</th>
<th>CPLEX time (sec)</th>
<th>10 segments BP total time (sec)</th>
<th>CPLEX time (sec)</th>
<th>50 segments BP total time (sec)</th>
<th>CPLEX time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>44.4</td>
<td>7.2</td>
<td>92.8</td>
<td>9.0</td>
<td>313.5</td>
<td>56.1</td>
</tr>
<tr>
<td>2</td>
<td>45.5</td>
<td>1.4</td>
<td>145.4</td>
<td>2.5</td>
<td>614.9</td>
<td>3.8</td>
</tr>
<tr>
<td>3</td>
<td>88.1</td>
<td>2.0</td>
<td>132.3</td>
<td>7.9</td>
<td>472.9</td>
<td>6.2</td>
</tr>
<tr>
<td>4</td>
<td>375.0</td>
<td>29.0</td>
<td>293.5</td>
<td>60.1</td>
<td>548.8</td>
<td>182.4</td>
</tr>
<tr>
<td>5</td>
<td>172.8</td>
<td>8.4</td>
<td>58.2</td>
<td>10.7</td>
<td>507.7</td>
<td>31.5</td>
</tr>
<tr>
<td>6</td>
<td>136.9</td>
<td>27.3</td>
<td>410.6</td>
<td>38.7</td>
<td>322.6</td>
<td>59.0</td>
</tr>
<tr>
<td>7</td>
<td>37.2</td>
<td>3.2</td>
<td>42.7</td>
<td>4.1</td>
<td>201.0</td>
<td>6.7</td>
</tr>
<tr>
<td>8</td>
<td>73.7</td>
<td>7.0</td>
<td>85.3</td>
<td>13.8</td>
<td>506.3</td>
<td>33.6</td>
</tr>
<tr>
<td>9</td>
<td>63.2</td>
<td>4.0</td>
<td>120.3</td>
<td>5.6</td>
<td>183.2</td>
<td>12.3</td>
</tr>
<tr>
<td>10</td>
<td>35.7</td>
<td>5.2</td>
<td>60.8</td>
<td>3.4</td>
<td>120.3</td>
<td>4.7</td>
</tr>
<tr>
<td>Avg</td>
<td>107.3</td>
<td>9.5</td>
<td>145.5</td>
<td>15.6</td>
<td>379.1</td>
<td>39.6</td>
</tr>
</tbody>
</table>
Table 4-11. CFLFD with linear revenue functions: 30 facilities, 60 customers, $\tau = 1.2$

<table>
<thead>
<tr>
<th>Exp</th>
<th>Cols</th>
<th>Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>CPLEX time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1071</td>
<td>1</td>
<td>9.9</td>
<td>9.9</td>
<td>3600(6.2)</td>
</tr>
<tr>
<td>2</td>
<td>964</td>
<td>7</td>
<td>9.5</td>
<td>15.1</td>
<td>3600(7.0)</td>
</tr>
<tr>
<td>3</td>
<td>988</td>
<td>1</td>
<td>6.9</td>
<td>6.9</td>
<td>3600(6.1)</td>
</tr>
<tr>
<td>4</td>
<td>928</td>
<td>1</td>
<td>10.3</td>
<td>10.3</td>
<td>3600(5.0)</td>
</tr>
<tr>
<td>5</td>
<td>1007</td>
<td>1</td>
<td>12.7</td>
<td>12.7</td>
<td>3600(1.2)</td>
</tr>
<tr>
<td>6</td>
<td>1008</td>
<td>1</td>
<td>9.3</td>
<td>9.3</td>
<td>3600(8.1)</td>
</tr>
<tr>
<td>7</td>
<td>1044</td>
<td>1</td>
<td>8.0</td>
<td>8.0</td>
<td>3600(4.2)</td>
</tr>
<tr>
<td>8</td>
<td>925</td>
<td>3</td>
<td>5.6</td>
<td>5.0</td>
<td>3600(6.1)</td>
</tr>
<tr>
<td>9</td>
<td>1176</td>
<td>31</td>
<td>8.9</td>
<td>37.2</td>
<td>3600(8.2)</td>
</tr>
<tr>
<td>10</td>
<td>1116</td>
<td>7</td>
<td>10.0</td>
<td>13.2</td>
<td>3600(1.3,2)</td>
</tr>
<tr>
<td>Avg</td>
<td>1022.7</td>
<td>5.4</td>
<td>9.1</td>
<td>12.8</td>
<td>3600(7.1)</td>
</tr>
</tbody>
</table>

Table 4-12. GAPFJ with nonlinear revenue functions: 5 facilities, 10 customers, $\tau = 1.2$

<table>
<thead>
<tr>
<th>Exp</th>
<th>Cols</th>
<th>Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>BARON time (sec)</th>
<th>Cols</th>
<th>Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>BARON time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>55</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>1123.3</td>
<td>54</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>100.0</td>
</tr>
<tr>
<td>2</td>
<td>58</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>1864.6</td>
<td>57</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>112.2</td>
</tr>
<tr>
<td>3</td>
<td>49</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>1036.6</td>
<td>50</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>86.1</td>
</tr>
<tr>
<td>4</td>
<td>65</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>3600(29.1)</td>
<td>64</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>63.7</td>
</tr>
<tr>
<td>5</td>
<td>62</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>2089.3</td>
<td>63</td>
<td>1</td>
<td>0.2</td>
<td>0.2</td>
<td>77.9</td>
</tr>
<tr>
<td>6</td>
<td>62</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>1636.0</td>
<td>63</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>101.9</td>
</tr>
<tr>
<td>7</td>
<td>64</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>1744.5</td>
<td>66</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>245.1</td>
</tr>
<tr>
<td>8</td>
<td>63</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>2539.7</td>
<td>65</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>65.8</td>
</tr>
<tr>
<td>9</td>
<td>66</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>1741.1</td>
<td>60</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>182.8</td>
</tr>
<tr>
<td>10</td>
<td>64</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>897.8</td>
<td>67</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>55.6</td>
</tr>
<tr>
<td>Avg</td>
<td>60.8</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>1827.3(3.1)</td>
<td>60.9</td>
<td>1</td>
<td>0.1</td>
<td>0.1</td>
<td>109.1</td>
</tr>
</tbody>
</table>
Table 4-13. GAPFJ with nonlinear revenue functions: 5 facilities, 15 customers, $\tau = 1.2$

<table>
<thead>
<tr>
<th>Exp</th>
<th>Cols</th>
<th>Nodes</th>
<th>Concave Root time (sec)</th>
<th>BARON time (sec)</th>
<th>Convex Root time (sec)</th>
<th>BARON time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>101</td>
<td>1</td>
<td>0.2</td>
<td>3600(^{(61,8)})</td>
<td>96</td>
<td>0.3</td>
</tr>
<tr>
<td>2</td>
<td>106</td>
<td>1</td>
<td>0.3</td>
<td>3600(^{(62,12)})</td>
<td>112</td>
<td>0.4</td>
</tr>
<tr>
<td>3</td>
<td>125</td>
<td>1</td>
<td>0.5</td>
<td>3600(^{(62,12)})</td>
<td>121</td>
<td>0.5</td>
</tr>
<tr>
<td>4</td>
<td>125</td>
<td>1</td>
<td>0.6</td>
<td>3600(^{(62,10)})</td>
<td>118</td>
<td>0.3</td>
</tr>
<tr>
<td>5</td>
<td>119</td>
<td>1</td>
<td>0.4</td>
<td>3600(^{(61,12)})</td>
<td>115</td>
<td>0.3</td>
</tr>
<tr>
<td>6</td>
<td>121</td>
<td>1</td>
<td>0.2</td>
<td>3600(^{(61,11)})</td>
<td>124</td>
<td>0.3</td>
</tr>
<tr>
<td>7</td>
<td>110</td>
<td>1</td>
<td>0.3</td>
<td>3600(^{(64,15)})</td>
<td>112</td>
<td>0.2</td>
</tr>
<tr>
<td>8</td>
<td>109</td>
<td>1</td>
<td>0.3</td>
<td>3600(^{(64,12)})</td>
<td>114</td>
<td>0.5</td>
</tr>
<tr>
<td>9</td>
<td>114</td>
<td>1</td>
<td>0.5</td>
<td>3600(^{(63,11)})</td>
<td>115</td>
<td>0.6</td>
</tr>
<tr>
<td>10</td>
<td>98</td>
<td>1</td>
<td>0.2</td>
<td>3600(^{(62,10)})</td>
<td>117</td>
<td>0.3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Avg 112.8</td>
<td>0.4</td>
<td>3600(^{(62,11)})</td>
<td>114.4</td>
</tr>
</tbody>
</table>

Table 4-14. GAPFJ with nonlinear revenue functions: 5 facilities, 25 customers, $\tau = 1.2$

<table>
<thead>
<tr>
<th>Exp</th>
<th>Cols</th>
<th>Nodes</th>
<th>Concave Root time (sec)</th>
<th>BARON time (sec)</th>
<th>Convex Root time (sec)</th>
<th>BARON time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>267</td>
<td>1</td>
<td>6.0</td>
<td>3600(^{(74,33)})</td>
<td>255</td>
<td>1.1</td>
</tr>
<tr>
<td>2</td>
<td>274</td>
<td>1</td>
<td>86.7</td>
<td>3600(^{(65,99)})</td>
<td>286</td>
<td>66.8</td>
</tr>
<tr>
<td>3</td>
<td>246</td>
<td>1</td>
<td>13.7</td>
<td>3600(^{(74,35)})</td>
<td>261</td>
<td>2.2</td>
</tr>
<tr>
<td>4</td>
<td>293</td>
<td>1</td>
<td>64.9</td>
<td>3600(^{(72,25)})</td>
<td>258</td>
<td>8.4</td>
</tr>
<tr>
<td>5</td>
<td>333</td>
<td>11</td>
<td>18.2</td>
<td>3600(^{(76,37)})</td>
<td>345</td>
<td>31.1</td>
</tr>
<tr>
<td>6</td>
<td>248</td>
<td>1</td>
<td>4.2</td>
<td>3600(^{(-,-)})</td>
<td>272</td>
<td>38.3</td>
</tr>
<tr>
<td>7</td>
<td>262</td>
<td>1</td>
<td>6.1</td>
<td>3600(^{(68,18)})</td>
<td>252</td>
<td>1.9</td>
</tr>
<tr>
<td>8</td>
<td>235</td>
<td>1</td>
<td>2.8</td>
<td>3600(^{(72,29)})</td>
<td>237</td>
<td>1.2</td>
</tr>
<tr>
<td>9</td>
<td>222</td>
<td>1</td>
<td>3.8</td>
<td>3600(^{(-,-)})</td>
<td>221</td>
<td>2.9</td>
</tr>
<tr>
<td>10</td>
<td>256</td>
<td>1</td>
<td>21.1</td>
<td>3600(^{(-,-)})</td>
<td>262</td>
<td>11.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Avg 263.6</td>
<td>2.0</td>
<td>264.9</td>
<td>1.6</td>
</tr>
</tbody>
</table>

69
Table 4-15. GAPFJ with linear revenue functions: 30 facilities, 90 customers, $\tau = 1.2$

<table>
<thead>
<tr>
<th>Exp</th>
<th>Cols</th>
<th>Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>CPLEX Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1130</td>
<td>1</td>
<td>79.0</td>
<td>89.4</td>
<td>10.0</td>
</tr>
<tr>
<td>2</td>
<td>1052</td>
<td>3</td>
<td>99.4</td>
<td>160.1</td>
<td>60.1</td>
</tr>
<tr>
<td>3</td>
<td>1109</td>
<td>1</td>
<td>89.0</td>
<td>104.8</td>
<td>15.5</td>
</tr>
<tr>
<td>4</td>
<td>1057</td>
<td>3</td>
<td>116.1</td>
<td>193.1</td>
<td>76.7</td>
</tr>
<tr>
<td>5</td>
<td>1053</td>
<td>3</td>
<td>91.4</td>
<td>97.8</td>
<td>6.0</td>
</tr>
<tr>
<td>6</td>
<td>1075</td>
<td>3</td>
<td>39.1</td>
<td>43.0</td>
<td>3.6</td>
</tr>
<tr>
<td>7</td>
<td>1124</td>
<td>3</td>
<td>108.5</td>
<td>129.5</td>
<td>20.6</td>
</tr>
<tr>
<td>8</td>
<td>1053</td>
<td>3</td>
<td>33.8</td>
<td>76.2</td>
<td>41.9</td>
</tr>
<tr>
<td>9</td>
<td>1101</td>
<td>1</td>
<td>55.1</td>
<td>107.8</td>
<td>52.3</td>
</tr>
<tr>
<td>10</td>
<td>1120</td>
<td>1</td>
<td>133.2</td>
<td>192.3</td>
<td>58.4</td>
</tr>
<tr>
<td>Avg</td>
<td>1087.4</td>
<td>2.2</td>
<td>84.5</td>
<td>119.4</td>
<td>34.5</td>
</tr>
</tbody>
</table>
CHAPTER 5
GAPFD HEURISTIC WITH ASYMPTOTIC PERFORMANCE GUARANTEES

In this chapter we consider the profit-maximizing GAPFD that requires the assignment of customers with flexible demand to available capacitated resources (facilities). As discussed in Chapter 3, this class of problems finds application in a wide range of practical settings. Related problems in sales and advertising planning involve tradeoffs between revenue generation and resource constraints and costs. In sales force planning contexts, for example, the sales force serves as a set of resources, where each salesperson has a limited amount of time and/or effort that they can allocate to customers. It is often the case that the greater the amount of effort a salesperson allocates to a given customer, the greater the return from that customer in terms of sales. The planning phase therefore involves determining the assignment of sales force to customers and the degree of effort a salesperson should devote to each assigned customer in order to maximize the total return from customers (or expected return, when the relationship between effort and sales is not deterministic). This sales setting may be interpreted more generally as applying to a set of available marketing instruments, where an allocation of capacity-constrained marketing instruments to customers must be determined in order to maximize profit.

Since the GAPFD generalizes the GAP, it is clearly $\mathcal{NP}$-Hard. Furthermore, since the feasibility problem associated with the GAP is $\mathcal{NP}$-Complete, it is clear that the feasibility problem associated with the GAPFD is $\mathcal{NP}$-Complete as well. We therefore develop a customized family of heuristics, and show that this class of heuristics is asymptotically feasible and optimal with probability one as the number of customers goes to infinity under a very broad probabilistic model for the problem parameters. Our heuristics are in the same spirit as certain heuristics that have been developed for the GAP by Martello and Toth [64] and Romeijn and Romero Morales [79]. In particular, given a vector of multipliers (each corresponding to a facility), a weight function is defined
to measure the pseudo-profit of assigning a customer to a facility. This weight function is then used to judiciously determine (i) the order in which to assign the customers, (ii) the facility to which each customer should be assigned, and (iii) an appropriate customer demand fulfillment level. In addition, these functions motivate improvement heuristics that are essential in order to be able to derive attractive performance guarantees for the heuristic. The main result of this chapter is the development of a heuristic that is asymptotically feasible and optimal with probability one under a very general stochastic model of the problem parameters. Due to the nature of the GAPFD, our approach for obtaining such guarantees is, particularly for the most general version of our model, significantly different from approaches used in case of the GAP. Specifically, we rely heavily both on the solution to a suitable perturbation of the GAPFD and on carefully designed solution improvement techniques. Thus, in addition to contributing to the literature on applied optimization in operations, we also provide new techniques for algorithm development and asymptotic analysis for combinatorial optimization problems. As our computational tests show, our heuristic solution approach is able to find optimal or near-optimal solutions with very limited computational effort for a broad range of problem dimensions.

The remainder of this chapter is organized as follows. Section 5.1 provides a number of important structural results; these results both motivate a class of heuristics and enable us to derive associated performance guarantees in Section 5.2. Section 5.3 discusses approaches to further improve the heuristic methods we propose, and in Section 5.4 we present the results of our computational study, which validate the effectiveness of our proposed methods.

### 5.1 Model Analysis

The heuristic proposed in this chapter is developed with respect to the GAPFD presented in Chapter 3 with linear revenue functions. That is, per unit of customer demand fulfillment a revenue of $r_{ij}$ is accrued. Moreover, we assume that $J_i = J$ ($i \in I$).
Specifically, the variant of the GAPFD introduced in Chapter 3 studied in this chapter is given by:

$$\text{maximize } \sum_{i \in I} \sum_{j \in J} p_{ij} x_{ij} + \sum_{i \in I} \sum_{j \in J} r_{ij} v_{ij} \quad (5-1)$$

subject to

$$\sum_{j \in J} a_{ij} x_{ij} + \sum_{j \in J} v_{ij} \leq b_i \quad i \in I \quad (5-2)$$

$$\sum_{i \in I} x_{ij} = 1 \quad j \in J \quad (5-3)$$

$$v_{ij} \geq \ell_{ij} x_{ij} \quad i \in I; j \in J \quad (5-4)$$

$$v_{ij} \leq u_{ij} x_{ij} \quad i \in I; j \in J \quad (5-5)$$

$$x_{ij} \in \{0, 1\} \quad i \in I; j \in J. \quad (5-6)$$

Note that the assumption that the unit resource consumption coefficients are equal to one can be made without loss of generality. Moreover, we will assume without loss of generality that the unit revenues are nonnegative. In principle, we allow the fixed profit and resource consumption coefficients to be either positive or negative. However, in most real-life applications we should expect these coefficients to be nonnegative. Finally, note that without loss of generality we could assume that $\ell_{ij} = 0$ for all $i \in I$ and $j \in J$ by appropriately modifying the fixed profit and resource consumption coefficients. However, for clarity of interpretation of our model, algorithms, and results we will allow for positive values of these lower bounds on the customer demand fulfillment levels.

The solution approach that we will develop and analyze in this chapter is a class of heuristics that is inspired by the Lagrange relaxation of a reformulation of the LP-relaxation of (P). In particular, as we will show below, the optimization problem (LP) that is obtained by replacing the binary constraints (5–6) by nonnegativity constraints

$$x_{ij} \geq 0 \quad i \in I; j \in J \quad (5-6')$$

...
is equivalent to the problem

\[
\text{maximize } \sum_{i \in I} \sum_{j \in J} (p_{ij} + r_{ij} u_{ij}) s_{ij} + \sum_{i \in I} \sum_{j \in J} (p_{ij} + r_{ij} \ell_{ij}) t_{ij} \quad (5-7)
\]

subject to

\[
\sum_{j \in J} (a_{ij} + u_{ij}) s_{ij} + \sum_{j \in J} (a_{ij} + \ell_{ij}) t_{ij} \leq b_i \quad i \in I \quad (5-8)
\]

\[
\sum_{i \in I} (s_{ij} + t_{ij}) = 1 \quad j \in J \quad (5-9)
\]

\[
s_{ij}, t_{ij} \geq 0 \quad i \in I; j \in J. \quad (5-10)
\]

**Theorem 2.** The optimization problems (LP) and (LP') are equivalent.

**Proof.** First note that we may modify (LP) by explicitly introducing (nonnegative) surplus and slack variables to constraints (5-4) and (5-5). For convenience, we will scale these so that they are expressed as a fraction of the width of the size range of the corresponding assignment. In other words, constraints (5-4) and (5-5) are replaced by

\[
v_{ij} - (u_{ij} - \ell_{ij}) s_{ij} = \ell_{ij} x_{ij} \quad i \in I; j \in J \quad (5-4')
\]

\[
v_{ij} + (u_{ij} - \ell_{ij}) t_{ij} = u_{ij} x_{ij} \quad i \in I; j \in J \quad (5-5')
\]

\[
s_{ij}, t_{ij} \geq 0 \quad i \in I; j \in J. \quad (5-11)
\]

It is easy to see that this reformulation is valid even if the width of the size range of an assignment is 0, i.e., if $\ell_{ij} = u_{ij}$. Now subtracting constraints (5-4') from (5-5') yields

\[
(u_{ij} - \ell_{ij}) x_{ij} = (u_{ij} - \ell_{ij}) (s_{ij} + t_{ij}) \quad i \in I; j \in J
\]

so that we can set, without loss of generality,

\[
x_{ij} = s_{ij} + t_{ij} \quad i \in I; j \in J. \quad (5-12)
\]

Moreover, multiplying constraints (5-4') and (5-5') by $u_{ij}$ and $\ell_{ij}$ respectively yields
\[ \ell_{ij} u_{ij} x_{ij} + u_{ij} (u_{ij} - \ell_{ij}) s_{ij} = u_{ij} v_{ij} \quad i \in I; \ j \in J \quad (5-4^\prime) \]

\[ \ell_{ij} u_{ij} x_{ij} - \ell_{ij} (u_{ij} - \ell_{ij}) t_{ij} = \ell_{ij} v_{ij} \quad i \in I; \ j \in J. \quad (5-5^\prime) \]

Subtracting \((5-5^\prime)\) from \((5-4^\prime)\), we obtain

\[ (u_{ij} - \ell_{ij}) (u_{ij} s_{ij} + \ell_{ij} t_{ij}) = (u_{ij} - \ell_{ij}) v_{ij} \quad i \in I; \ j \in J \]

so that we can set

\[ v_{ij} = u_{ij} s_{ij} + \ell_{ij} t_{ij} \quad i \in I; \ j \in J. \quad (5-13) \]

Notice that the non-negativity of the slack and surplus variables by \((5-11)\) along with \((5-12)\) and \((5-13)\) implies that \((5-10)\) is sufficient to ensure all non-negativity conditions in \((LP)\) are also satisfied in \((LP^\prime)\). Finally, substituting \((5-12)\) and \((5-13)\) into the objective \((5-1)\) and constraints \((5-2)\) and \((5-3)\) of \((P)\) yields the objective \((5-7)\) as well as constraints \((5-8)\) and \((5-9)\) of \((LP^\prime)\).

Next, denote the (nonnegative) dual multipliers of the capacity constraints \((5-8)\) by \(\lambda_i\) \((i \in I)\) and the (free) dual multipliers of the assignment constraints \((5-9)\) by \(\mu_j\) \((j \in J)\).

The dual \((D^\prime)\) of \((LP^\prime)\) is then given by

\[
\text{minimize} \sum_{i \in I} \lambda_i b_i + \sum_{j \in J} \mu_j \\
\text{subject to} \\
\mu_j \geq p_{ij} - \lambda_i a_{ij} + (r_{ij} - \lambda_i) \ell_{ij} \quad i \in I; \ j \in J \quad (5-14) \\
\mu_j \geq p_{ij} - \lambda_i a_{ij} + (r_{ij} - \lambda_i) u_{ij} \quad i \in I; \ j \in J \quad (5-15) \\
\lambda_i \geq 0 \quad i \in I \\
\mu_j \text{ free} \quad j \in J.
\]
The following theorem derives a convenient and insightful expression for the optimal value to both (LP') and (D') as a function of the dual multipliers $\lambda_i$ ($i \in \mathcal{I}$) of the capacity constraints (5–8) only.

**Theorem 3.** The common optimal value of (LP') and (D') can be expressed as

$$\min_{\lambda \geq 0} L(\lambda)$$

where

$$L(\lambda) = \sum_{j \in \mathcal{J}} \max_{i \in \mathcal{I}} f_\lambda(i, j) + \sum_{i \in \mathcal{I}} \lambda_i b_i$$

and where

$$f_\lambda(i, j) = \begin{cases} p_{ij} - \lambda_i a_{ij} + (r_{ij} - \lambda_i) u_{ij} & \text{if } \lambda_i \leq r_{ij} \\ p_{ij} - \lambda_i a_{ij} + (r_{ij} - \lambda_i) \ell_{ij} & \text{if } \lambda_i > r_{ij} \end{cases}$$

**Proof.** From constraints (5–14) and (5–15) we obtain that, without loss of optimality, the dual variables $\mu_j$ can be chosen as

$$\mu_j = \max_{i \in \mathcal{I}} \max \{ p_{ij} - \lambda_i a_{ij} + (r_{ij} - \lambda_i) \ell_{ij}, p_{ij} - \lambda_i a_{ij} + (r_{ij} - \lambda_i) u_{ij} \} \quad (5–16)$$

Now note that the inner maximum in (5–16) is attained by the first argument if $\lambda_i \geq r_{ij}$ and by the second argument if $\lambda_i \leq r_{ij}$. This implies that we in fact have

$$\mu_j = \max_{i \in \mathcal{I}} f_\lambda(i, j) \quad j \in \mathcal{J} \quad (5–17)$$

which yields the desired result. \qed

It is useful to introduce some terminology with respect to a feasible solution $(s, t)$ to (LP'). Consider some customer $j$. If $x_{ij} = s_{ij} + t_{ij} = 1$ for some facility $i$ we say that customer $j$ is assigned to facility $i$ and furthermore, customer $j$ is referred to as a *non-split* customer. Similarly, if $0 < x_{ij} = s_{ij} + t_{ij} < 1$ for some facility $i$ we say that the assignment of customer $j$ to facility $i$ is *fractional*, and customer $j$ is referred to as a *split customer.*
More formally, we define the set

$$\mathcal{F} = \{(i, j) : 0 < x_{ij} < 1\} = \{(i, j) : 0 < s_{ij} + t_{ij} < 1\}$$

of fractional assignments, and the set

$$\mathcal{S} = \{j : \exists i \text{ such that } (i, j) \in \mathcal{F}\}$$

of split customers. Furthermore, we will say that an assignment $(i, j)$ such that $s_{ij} > 0$ and $t_{ij} = 0$ is executed at its upper bound, while an assignment $(i, j)$ such that $s_{ij} = 0$ and $t_{ij} > 0$ is executed at its lower bound. The set

$$\mathcal{Q} = \{(i, j) : s_{ij} > 0 \text{ and } t_{ij} > 0\}$$

then consists of the facility/customer pairs executed strictly between their bounds. Finally,

$$\mathcal{C} = \left\{ i : \sum_{j \in \mathcal{J}} a_{ij} x_{ij} + \sum_{j \in \mathcal{J}} v_{ij} = b_i \right\} = \left\{ i : \sum_{j \in \mathcal{J}} (a_{ij} + u_{ij}) s_{ij} + \sum_{j \in \mathcal{J}} (a_{ij} + \ell_{ij}) t_{ij} = b_i \right\}$$

is the set of facilities that operate at full-capacity.

The following theorem establishes a close relationship between an optimal solution to (D') and the corresponding primal optimal solution, provided that the latter is unique.

**Theorem 4.** Suppose that $(LP')$ is feasible and that the optimal (basic) solution to $(LP')$, say $(s^*, t^*)$, is unique. Furthermore, let $\lambda^*$ be an associated complementary optimal solution to (D'). The primal and dual solutions then satisfy the following properties.

(i) Let $j \in \mathcal{S}$ be a split customer. Then there exists a facility $i'$ such that

$$f_{\lambda^*}(i', j) = \max_{i \in \mathcal{I}, i \neq i'} f_{\lambda^*}(i, j).$$

(ii) Let $j \notin \mathcal{S}$ be a non-split customer. Then it is assigned to facility $i'$ if and only if

$$f_{\lambda^*}(i', j) = \max_{i \in \mathcal{I}} f_{\lambda^*}(i, j)$$

and

$$f_{\lambda^*}(i', j) > \max_{i \in \mathcal{I}, i \neq i'} f_{\lambda^*}(i, j).$$
(iii) Let \( j \not\in S \) be a non-split customer that is assigned to facility \( i' \). Then

\[
s_{i'j}^* = 1 - t_{i'j}^* \begin{cases} 
0 & \text{if } \lambda_{i'}^* > r_{i'j} \\
\in [0, 1] & \text{if } \lambda_{i'}^* = r_{i'j} \\
1 & \text{if } \lambda_{i'}^* < r_{i'j}.
\end{cases}
\]

Proof. First, note that uniqueness of the optimal solution \((s^*, t^*)\) implies that \((D')\) is nondegenerate. To simplify notation, denote \( x^* = s^* + t^* \) (recalling the relationship given in (5–12)).

(i) Let \( j \in S \) be a split customer. This implies that there exist (at least) two facilities, say \( i' \) and \( i'' \), such that \( x_{i'j}, x_{i''j} > 0 \). By the definition of \( x^* \), complementary slackness now implies that for both \( i' \) and \( i'' \), at least one of the corresponding dual constraints (5–14) and (5–15) is binding. This, in turn, implies that for this customer the maximum in equation (5–17) is attained for both facilities \( i' \) and \( i'' \), yielding claim (i).

(ii) Let \( j \not\in S \) be a non-split customer. This implies that there exists only a single facility, say \( i' \), such that \( x_{i'j} > 0 \) (in fact, \( x_{i'j} = 1 \)). By complementary slackness and the nondegeneracy of the dual solution, this means that for all facilities except \( i' \) both corresponding dual constraints (5–14) and (5–15) are nonbinding. This, in turn, implies that for this customer the maximum in equation (5–17) is attained for only facility \( i' \), yielding claim (ii).

(iii) Let \( j \not\in S \) be a non-split customer that is assigned to facility \( i' \), so that \( x_{i'j}^* = 1 \). First, recall that \( s_{ij} \) is the primal variable associated with (5–14) and \( t_{ij} \) is the primal variable associated with (5–15). Now by complementary slackness and dual nondegeneracy we have that \( s_{i'j}^* > 0 \) and \( t_{i'j}^* = 0 \) if and only if \( \lambda_{i'}^* > r_{i'j} \), \( s_{i'j}^* = 0 \) and \( t_{i'j}^* > 0 \) if and only if \( \lambda_{i'}^* < r_{i'j} \), and \( s_{i'j}^* > 0 \) and \( t_{i'j}^* > 0 \) if and only if \( \lambda_{i'}^* = r_{i'j} \). Together with (5–12) this yields the desired result.

\[ \square \]

It is interesting to see what Theorem (4) implies in terms of the \((x, v)\) variables in our original (LP) formulation.

**Corollary 1.** Suppose that \((LP')\) is feasible and that the optimal (basic) solution to \((LP')\), say \((s^*, t^*)\), is unique. Furthermore, let \( \lambda^* \) be an associated complementary optimal solution to \((D')\). Then there exists an optimal solution \((x^*, v^*)\) to \((LP)\) that satisfies the following property. If \( j \not\in S \) is a non-split customer that is assigned to facility \( i \) in the
optimal solution to (LP), then

\[
v_{ij}^* = \begin{cases} 
\ell_{ij} & \text{if } \lambda_{ij}^* > r_{ij} \\
\in [\ell_{ij}, u_{ij}] & \text{if } \lambda_{ij}^* = r_{ij} \\
u_{ij} & \text{if } \lambda_{ij}^* < r_{ij}.
\end{cases}
\]

Proof. The result follows immediately from Theorems 2 and 4. \hfill \Box

Remarks:

1. Note that if \( \ell_{ij} = u_{ij} \) for at least one pair \((i, j)\), the optimal solution to (LP) cannot be unique. However, in this case we can arbitrarily set \( t_{ij} \equiv 0 \) for such pairs, which would allow uniqueness of the optimal solution to (LP).

2. Clearly, uniqueness of the optimal solution to (LP), even with the modification given in Remark 1, cannot be guaranteed for all instances. However, in Section 5.2 we will introduce a stochastic model for problem instances of (P) under which this can be guaranteed with probability one.

In Section 5.2 we will use the insights of Theorems 3 and 4 to develop a heuristic for solving the GAPFD. We will, however, first derive an important result characterizing the nature of basic feasible solutions to (LP) that will later prove significant in the average case performance analysis of that heuristic. To arrive at this result we note that the number of nonzero variables in a basic feasible solution is bounded by the number of equality constraints in (LP). We then count the number of basic variables using the set descriptions defined previously. This analysis provides an important inequality which is used in Corollary 2 to arrive at a bound on the number of split customers and customers assigned between their bounds in an optimal solution to (LP).

Theorem 5. Let \((s, t)\) be a basic feasible solution to (LP). Then the total number of assignments that are either fractional or strictly between their bounds is bounded by the total number of split customers plus the total number of facilities operating at full capacity, i.e.,

\[ |\mathcal{F}| + |\mathcal{Q}| \leq |\mathcal{S}| + |\mathcal{C}|. \]
\textit{Proof.} The optimization problem (LP') has $2|\mathcal{I}||\mathcal{J}|$ variables and $|\mathcal{I}| + |\mathcal{J}|$ equality constraints. The total number of variables which are nonzero in a basic feasible solution is therefore no larger than $|\mathcal{I}| + |\mathcal{J}|$. Now observe that there are

- $|\mathcal{J}| - |\mathcal{S}| + |\mathcal{F}| + |\mathcal{Q}|$ non-zero components in $(s,t)$;
- $|\mathcal{I}| - |\mathcal{C}|$ non-zero slack variables associated with constraints (5–8);

This yields

$$|\mathcal{I}| + |\mathcal{J}| \geq (|\mathcal{J}| - |\mathcal{S}| + |\mathcal{F}| + |\mathcal{Q}|) + (|\mathcal{I}| - |\mathcal{C}|)$$

which yields the desired result. \hfill \Box

\textbf{Corollary 2.} Let $(s,t)$ be a basic feasible solution to (LP'). Then the total number of customers that are either split or executed strictly between their bounds is bounded by the number of facilities, i.e.,

$$|\mathcal{S}| + |\mathcal{Q}| \leq |\mathcal{I}|.$$

\textit{Proof.} Note that each split customer has at least two corresponding fractional assignment variables, so that $|\mathcal{F}| \geq 2|\mathcal{S}|$. The result then follows directly from Theorem 5 and the fact that $|\mathcal{C}| \leq |\mathcal{I}|$. \hfill \Box

\section*{5.2 An Asymptotically Optimal Heuristic}

\subsection*{5.2.1 Development of the Heuristic}

There is an attractive intuitive interpretation of the result of Theorem 3 by noting that we can interpret the value of the dual variable $\lambda_i$ as a unit cost of capacity of facility $i$. Then, note that we can view $p_{ij} - \lambda_i a_{ij}$ as a fixed pseudo-profit that is received if customer $j$ is assigned to facility $i$, regardless of its level. Next, we can view the difference between the actual corresponding unit revenue $r_{ij}$ and the cost $\lambda_i$ of using a unit of capacity of facility $i$ as a unit pseudo-profit that is received if customer $j$ is assigned to facility $i$. The sign of the pseudo-profit then indicates the level at which a customer should be assigned: if the unit pseudo-profit is positive, customer $j$ (if assigned to facility $i$) is executed at its upper bound $u_{ij}$, yielding a total pseudo-profit of $p_{ij} - \lambda_i a_{ij} + (r_{ij} - \lambda_i)u_{ij}$. 80
Similarly, if the unit pseudo-profit is negative, customer \( j \) (if assigned to facility \( i \)) is executed at its lower bound \( \ell_{ij} \), yielding a total pseudo-profit of \( p_{ij} - \lambda_i a_{ij} + (r_{ij} - \lambda_i)\ell_{ij} \).

In summary, the function \( f_\lambda(i, j) \) can be viewed as a pseudo-profit associated with the assignment of customer \( j \) to facility \( i \) for a given vector of dual prices \( \lambda \).

We will use this interpretation to propose a heuristic for the GAPFD. That is, our heuristic will, to a large extent, assign customers according to the pseudo-profit function.

(Note that any nonnegative vector \( \lambda \) defines a distinct pseudo-profit function, so that we will in fact obtain a family of heuristics. However, we later show that the heuristic enjoys an attractive performance guarantee if we use an optimal dual solution to either the original problem or a perturbation thereof.) In particular, we will attempt to assign each customer to the facility that maximizes its pseudo-profit function and select the corresponding customer demand fulfillment level accordingly. More specifically, the most profitable facility for customer \( j \) is given by

\[
i_j = \arg \max_{i \in I_j} f_\lambda(i, j)
\]

where \( I_j \subseteq I \) is the set of facilities currently under consideration for customer \( j \). It is easy to see that, in general, assigning all customers \( j \) to their most profitable facility \( i_j \) at a size as described in the preceding paragraph cannot be expected to yield a feasible solution to the GAPFD. We therefore select the order in which the customers are assigned by considering not only the maximum pseudo-profit but also the second largest pseudo-profit for each customer. We define the difference between these two values:

\[
\rho_j = f_\lambda(i_j, j) - \max_{i' \in I_j \setminus \{i_j\}} f_\lambda(i', j)
\]

to be the desirability of assigning customer \( j \) to its most profitable facility. We then assign the customers to their most profitable facility in nonincreasing order of desirability, as long as it is feasible to do so.
Our heuristic proceeds in two phases. In the first, greedy phase of the heuristic, the set $\mathcal{U}$ keeps track of the set of customers that remain to be assigned. During the course of this phase (Steps 1–3), customers may be identified that can no longer be feasibly assigned. The set $\tilde{\mathcal{J}}$ of such customers will be handled in the second, improvement phase of the algorithm (Steps 4–7). Throughout the heuristic, $b'_i$ denotes the capacity remaining for facility $i$ ($i \in \mathcal{I}$).

We now formally present our heuristic as follows:

**Heuristic – Greedy phase**

**Step 0.** Set $\mathcal{U} = \mathcal{J}$, $\tilde{\mathcal{J}} = \emptyset$, $b'_i = b_i$ for $i \in \mathcal{I}$, and $\mathcal{I}_j = \mathcal{I}$. Set $x^G_{ij} = v^G_{ij} = 0$ for $i \in \mathcal{I}$; $j \in \mathcal{J}$.

**Step 1.** Let

$$i_j \in \arg\max_{i \in \mathcal{I}_j} f_\lambda(i, j) \quad \text{for } j \in \mathcal{U}$$

$$\rho_j = f_\lambda(i_j, j) - \max_{i' \in \mathcal{I}_j \setminus \{i_j\}} f_\lambda(i', j) \quad \text{for } j \in \mathcal{U}.$$ 

**Step 2.** Select $\hat{j} \in \arg\max_{j \in \mathcal{U}} \rho_j$, i.e., $\hat{j}$ is the customer to be assigned next (to facility $i_{\hat{j}}$).

If $a_{i_{\hat{j}}} + \ell_{i_{\hat{j}}} \leq b'_{i_{\hat{j}}}$ continue to Step 3. Otherwise, $a_{i_{\hat{j}}} + \ell_{i_{\hat{j}}} > b'_{i_{\hat{j}}}$, which means this assignment is not feasible; let $\mathcal{I}_{\hat{j}} = \{i : a_{ij} + \ell_{ij} \leq b'_i\}$. If $\mathcal{I}_{\hat{j}} = \emptyset$, set $\tilde{\mathcal{J}} = \mathcal{U}$ and STOP.

**Step 3.** Set

$$x^G_{i_{\hat{j}}j} = 1$$

$$v^G_{i_{\hat{j}}j} = \begin{cases} \min\{u_{i_{\hat{j}}j}, b'_i - a_{i_{\hat{j}}j}\} & \text{if } r_{i_{\hat{j}}j} > \lambda_{ij} \\ \ell_{i_{\hat{j}}j} & \text{if } r_{i_{\hat{j}}j} \leq \lambda_{ij} \end{cases}$$

$$b'_{i_{\hat{j}}} = b'_i - (v^G_{i_{\hat{j}}j} + a_{i_{\hat{j}}j}).$$

Let $\mathcal{U} = \mathcal{U} \setminus \{j\}$. If $\mathcal{U} \neq \emptyset$, return to Step 1; otherwise, STOP.
If the greedy phase of the heuristic ends with $\tilde{\mathcal{J}} = \emptyset$, then $(x^G, v^G)$ is a feasible solution to the GAPFD. Otherwise, we will continue the heuristic with an improvement phase. To distinguish the (partial) solution obtained at the end of the greedy phase from the solution delivered by the improvement phase we set $(x^H, v^H) = (x^G, v^G)$.

In the improvement phase, we reduce the size of some previously assigned customers to their corresponding lower bounds to free up capacity that can be used to assign any of the customers in $\tilde{\mathcal{J}}$. Realizing that the minimum amount of capacity that is required to assign customer $j$ to facility $i$ is $a'_{ij} \equiv a_{ij} + \ell_{ij}$, let $\bar{a}'$ be an upper bound on this value among all unassigned customers. Then, at least

$$\sum_{i \in \mathcal{I}} b'_i \geq \frac{1}{\bar{a}'} \sum_{i \in \mathcal{I}} b'_i - |\mathcal{I}|$$

customers can be accommodated within the remaining facility capacities. Thus, all customers in $\tilde{\mathcal{J}}$ can be assigned if the facilities have cumulative available capacity $\sum_{i \in \mathcal{I}} b'_i \geq (|\tilde{\mathcal{J}}| + |\mathcal{I}|) \bar{a}'$. Note that such an assignment can be found by arbitrarily assigning the customers in $\tilde{\mathcal{J}}$ to any facility that can feasibly accommodate it.

**Heuristic – Improvement phase**

**Step 4.** Let $\mathcal{A} = \{(i, j) : x^H_{ij} = 1 \text{ and } v^H_{ij} > \ell_{ij}\}$ and set $a' = \max_{(i,j) \in \mathcal{I} \times \tilde{\mathcal{J}}} (\ell_{ij} + a_{ij})$.

**Step 5.** Identify a set $\mathcal{A}' \subseteq \mathcal{A}$ with the property that

$$\sum_{i \in \mathcal{I}} b'_i + \sum_{(i,j) \in \mathcal{A}'} (v^H_{ij} - \ell_{ij}) \geq (|\tilde{\mathcal{J}}| + |\mathcal{I}|) \bar{a}' \tag{5–18}$$

and, in addition, $\mathcal{A}'$ is minimal in the sense that removing any element from it causes (5–18) to be violated. If such a set does not exist, set $\mathcal{A}' = \mathcal{A}$.

**Step 6.** Set

$$b'_i = b'_i + \sum_{j: (i,j) \in \mathcal{A}'} (v^H_{ij} - \ell_{ij}) \quad \text{for } i \in \mathcal{I}$$

$$v^H_{ij} = \ell_{ij} \quad \text{for } (i, j) \in \mathcal{A}'$$
Step 7. Attempt to identify a feasible solution to the GAPFD by (i) assigning and
determining customers and customer demand fulfillment levels for \( j \in \tilde{J} \), and
(ii) increasing customer demand fulfillment levels for assignments \((i,j) \in \mathcal{A}'\).
If this is successful, return the solution as \((x^H,v^H)\). Otherwise, the heuristic is
unable to find a feasible solution to the GAPFD.

In the remainder of this section we analyze a basic implementation of the improvement
phase of the heuristic where we identify an arbitrary set \( \mathcal{A}' \) in Step 5, and try to assign
customers in \( \tilde{J} \) in arbitrary order to facilities that can accommodate them in Step 7. In
Section 5.4 we will propose a more sophisticated implementation with guaranteed superior
behavior.

The following theorem establishes a close relationship between the solution that is
obtained by the greedy phase of the heuristic and a basic optimal solution to \((LP')\).

**Theorem 6.** Suppose that \((LP')\) is feasible and that the optimal (basic) solution to \((LP')\),
say \((s^*,t^*)\), is unique. If we choose \( \lambda \) in the heuristic equal to an associated optimal dual
vector \( \lambda^* \), we have for all non-split customers \( j \not\in S \) that the greedy phase of the heuristic
(i) assigns this customer to the same facility, say \( i_j \), as \((LP')\);
(ii) executes this customer at the same level as \((LP')\) provided \((i_j,j) \not\in Q\).

**Proof.** Theorem 4(i)–(ii) implies that, in the greedy phase of the heuristic, \( \rho_j > 0 \) for all
\( j \not\in S \) and \( \rho_j = 0 \) for \( j \in S \). Thus, Step 2 guarantees that all non-split customers are
considered before any split customers as long as the sets \( \mathcal{I}_j \) remain unchanged. Claim (i)
then immediately follows from the fact that the preferred assignments of the non-split
customers are all feasible. Next, Theorem 4(iii) implies that, in Step 3 of the greedy phase
of the heuristic, all customers \( j \) for which \((i,j) \not\in Q\) are executed at the same level as
in \((LP')\). (Note that the greedy phase does not necessarily execute customers for which
\((i,j) \in Q\) at the same level as in \((LP')\). This follows since, by complementary slackness,
\((i,j) \in Q\) is equivalent to \( \mu_j = f_i(i,j) = p_{ij} - \lambda_i a_{ij} \), i.e. the unit revenue \( r_{ij} - \lambda_i = 0 \),
which does not necessarily imply that the solution to (LP') makes this assignment at its lower bound, as the heuristic does.)

Theorem 6 states that, if we choose $\lambda$ in the heuristic equal to an optimal dual vector $\lambda^*$ of (LP'), the greedy phase of the heuristic starts by making the assignments of non-split customers in the solution to (LP'), with the only possible deviation being the customer demand fulfillment levels for those that are strictly between their bounds in that solution. This, together with Corollary 2, then implies that the total number of customers that are unassigned in the greedy phase or for which the assignment or demand fulfillment level differs from the solution to (LP') is no larger than the number of facilities, $|\mathcal{I}|$. The improvement phase of the heuristic is aimed at creating sufficient space to allow the assignment of any unassigned customers. Our goal in the next section is to use this result to derive strategies for choosing $\lambda$ for two very general stochastic models on the problem parameters such that the heuristic is asymptotically feasible and optimal with probability one as the number of customers increases. That is, as the number of customers increases, it is very unlikely that the heuristic is unable to find a feasible solution and, moreover, if it finds a feasible solution, its relative error will decline as the number of customers increases.

5.2.2 Average Case Analysis of the Heuristic

This section provides an analysis of the asymptotic behavior of our heuristic under a probabilistic model on the problem parameters that keeps the number of facilities, $|\mathcal{I}|$, fixed and lets the number of customers, $|\mathcal{J}|$, approach infinity. We propose a stochastic model for the GAPFD that is similar to the ones commonly used for the GAP and its extensions (see, e.g., Dyer and Frieze [30], Romeijn and Piersma [78], and Romeijn and Romero Morales [80]). In particular, we assume that each customer is characterized by a random vector of parameters $(P_j, R_j, A_j, L_j, D_j)$, where $P_j = (P_{1j}, \ldots, P_{|\mathcal{I}|j}), R_j =$

---

1 As a convention, we will denote parameters and solutions that are random variables by capital letters, while realizations will be denoted by lowercase letters.
Here $P_j$ and $R_j$ are the vectors of fixed profits and unit revenues for customer $j$, respectively. Furthermore, $A_j$ is the vector of fixed resource consumptions for customer $j$, while $L_j$ is the vector of lower bounds and $U_j \equiv L_j + D_j$ is the vector of upper bounds on the size of customer $j$. The vectors $(P_j, R_j, A_j, L_j, D_j)$ are assumed to be i.i.d. on the compact set $[P, \bar{P}]^{|I|} \times [R, \bar{R}]^{|I|} \times [A, \bar{A}]^{|I|} \times [L, \bar{L}]^{|I|} \times [D, \bar{D}]^{|I|}$, where the conditional distributions of $(P_j, R_j | A_j, L_j, D_j)$ are absolutely continuous. In addition, we have $R, L \geq 0$ so that both the demand requirement and the unit revenue accrued are non-negative. Furthermore, the difference between the upper and lower bound parameters is taken to be strictly positive, $D > 0$, to ensure there is a decision to be made with regard to the level at which a customer’s demand is satisfied. For convenience, we assume that $R L > -P$ so that the total profit associated with any feasible assignment is nonnegative. (Note that this assumption is mild since it is automatically satisfied if, for example, $P > 0$. Moreover, it can be made without loss of generality since we may add or subtract a constant value from all fixed profit coefficients without impacting the profit ranking of the solutions.) Furthermore, as is common in probabilistic models of this type, we allow for the accommodation of an increasing number of customers while the number of facilities remains constant by letting the capacity of facility $i$ grow linearly with the number of customers, i.e., we let $b_i = \beta_i |J|$ where $\beta_i$ is a positive constant ($i \in I$).

Finally, we wish to focus on problem instances that admit a feasible solution. Note that an instance of the GAPFD has a feasible solution if and only if the associated GAP with all requirements set to their minimum value $a_{ij} + \ell_{ij}$ ($i \in I; j \in J$) is feasible. This leads to the following assumption that we will impose on our probabilistic model:

Assumption 1.

$$\Delta \equiv \min_{\lambda \geq 0; \lambda^T e = 1} \left( \sum_{i \in I} \lambda_i \beta_i - E \left( \min_{i \in I} \lambda_i (A_{i1} + L_{i1}) \right) \right) > 0.$$
By Romeijn and Piersma [78, Theorem 3.2], this assumption ensures that an instance randomly generated according to our stochastic model is feasible with probability one as $|\mathcal{J}| \to \infty$. Note that this assumption is mild, since they also show that instances generated are asymptotically infeasible with probability one as $|\mathcal{J}| \to \infty$ if $\Delta < 0$.

In the remainder of this section, we will show that, for a suitably chosen strategy for the parameter $\lambda$, the heuristic will provide a feasible and optimal solution to the GAPFD with probability one as $|\mathcal{J}| \to \infty$. In particular, consider an instance of the GAPFD generated from the probabilistic model described above, and let $Z_{\mathcal{J}}^*$, $Z_{\mathcal{J}}^{\text{LP}}$, $Z_{\mathcal{J}}^\mathcal{G}$, and $Z_{\mathcal{J}}^\mathcal{H}$ denote its optimal solution value, the value of its LP-relaxation, and the value of the solution obtained by the greedy and the improvement phases of the heuristic, respectively. (Note that these values are random variables, and that we have explicitly recognized that they are a function of the number of customers, $|\mathcal{J}|$.) We then say that the heuristic is asymptotically feasible and optimal if

(i) the solution $(X^\mathcal{H}, V^\mathcal{H})$ produced by the heuristic is asymptotically feasible with probability one;

(ii) $\lim_{|\mathcal{J}| \to \infty} \left( \frac{Z_{\mathcal{J}}^* - Z_{\mathcal{J}}^\mathcal{H}}{Z_{\mathcal{J}}^*} \right) = 0$ with probability one.

Since, under our assumptions on the problem parameters, we have that, for any feasible instance of the GAPFD, $Z_{\mathcal{J}}^* \geq (R L + P)|\mathcal{J}|$ with $R L + P > 0$, the latter is equivalent to

(ii') $\lim_{|\mathcal{J}| \to \infty} \frac{1}{|\mathcal{J}|} \left( \frac{Z_{\mathcal{J}}^* - Z_{\mathcal{J}}^\mathcal{H}}{Z_{\mathcal{J}}^*} \right) = 0$ with probability one

which is the characterization of asymptotic optimality that we will use in the remainder of this chapter.

We will distinguish between two classes of instances of the GAPFD. The first class of instances is characterized by customer requirements that are facility-independent, i.e., $A_{ij} = A_{1j}$, $L_{ij} = L_{1j}$, and $D_{ij} = D_{1j}$ ($i \in \mathcal{I}; j \in \mathcal{J}$). For this class, we will show that the heuristic is asymptotically feasible and optimal with probability one if we choose $\lambda$ equal to an associated optimal dual vector $\lambda^*$. The second class of instances follows the general probabilistic model discussed earlier in this section. For this class, we will show that the
heuristic is asymptotically feasible and optimal with probability one if we choose $\lambda$ equal to an optimal dual vector of an appropriately perturbed instance of the GAPFD.

5.2.2.1 Facility-independent requirements

Recall that Theorem 6 establishes a strong connection between an optimal solution to (LP$'$) and the solution obtained by the greedy phase of the heuristic if the former is unique with respect to non-split customers and if we choose $\lambda$ equal to an associated optimal dual vector $\lambda^*$ of (LP$'$). This connection is employed to obtain asymptotic feasibility and optimality. Before we formally prove this result, however, we will first derive two useful preliminary results. The first result provides an intuitive characterization of an optimal solution to (LP$'$) under the proposed stochastic model. This result is common for models in which parameters are generated from absolutely continuous distributions (see [29, 30, 79]). It is presented here formally due to its significance in our asymptotic analysis.

**Lemma 3.** Under our stochastic model, if (LP$'$) is feasible, its optimal solution is unique with probability one.

**Proof.** A non-unique optimal solution to (LP$'$) exists only if the hyperplane representative of solutions with optimal profit

$$\sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} (p_{ij} + r_{ij}u_{ij}) s_{ij}^* + \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} (p_{ij} + r_{ij}f_{ij}) t_{ij}^*$$

intersects the feasible region at multiple points. Recall that the unit revenues $r_{ij}$ ($i \in \mathcal{I}; j \in \mathcal{J}$) and the fixed profits $p_{ij}$ ($i \in \mathcal{I}; j \in \mathcal{J}$) are generated from a joint distribution that, conditional on the values of the requirements parameters in the constraints, is absolutely continuous. Thus the probability that data generated by our stochastic model allows for multiple optimal solutions to (LP$'$) is zero, so that the optimal solution to (LP$'$) is unique with probability one. \qed
Lemma 4. When customer requirements are facility-independent, the aggregate capacity that is either unused or used for customer levels in excess of their lower bound increases linearly in $|\mathcal{J}|$ with probability one as $|\mathcal{J}| \to \infty$, in any feasible solution to (LP).

Proof. For convenience, denote the effective lower bound on an assignment by $A'_{ij} = A_{ij} + L_{ij}$. Since the customer requirements are facility-independent we have that $A'_{ij} = A'_{1j}$ ($i \in \mathcal{I}$). Then, for any assignment vector $x$ that is feasible to the LP-relaxation of the corresponding GAP we have

$$\frac{1}{|\mathcal{J}|} \sum_{i \in \mathcal{I}} \left( b_i - \sum_{j \in \mathcal{J}} A'_{ij} x_{ij} \right) = \sum_{i \in \mathcal{I}} \beta_i - \frac{1}{|\mathcal{J}|} \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} A'_{1j} x_{ij} = \sum_{i \in \mathcal{I}} \beta_i \left( \sum_{j \in \mathcal{J}} A'_{ij} \right) = \sum_{i \in \mathcal{I}} \beta_i - \frac{1}{|\mathcal{J}|} \sum_{j \in \mathcal{J}} A'_{1j}.$$ 

For the case of facility-independent requirements, Romeijn and Piersma [78] show that Assumption 1 is equivalent to the condition

$$E(A'_{1j}) < \sum_{i \in \mathcal{I}} \beta_i$$

so that, by the Central Limit Theorem,

$$\sum_{i \in \mathcal{I}} \beta_i - \frac{1}{|\mathcal{J}|} \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} A'_{ij} x_{ij} > 0 \quad \text{with probability one if } |\mathcal{J}| \to \infty$$

for any feasible relaxed assignment vector $x$. This yields the desired result. \qed

We are now ready to formally prove our first asymptotic feasibility and optimality result.

Theorem 7. Consider problem instances generated according to our stochastic model with, in addition, facility-independent customer requirements. Moreover, choose $\lambda$ in the heuristic equal to an associated optimal dual vector $\lambda^*$ of $(LP')$. Then the heuristic is asymptotically feasible and optimal.
Proof. Since we are only interested in a probabilistic and asymptotic feasibility guarantee, we may by Lemma 3 assume that the solution to (LP') is unique with respect to non-split customers. Then Theorem 6 says that the greedy phase of the heuristic assigns no more than \( |S| \) customers to a different facility or to no facility at all, and no more than \( |Q| \) customers to the same facility but at a different level than the optimal solution to (LP').

Again denoting the effective lower bound of a customer by \( A' = A + L \) with corresponding upper bound \( \bar{A}' = \bar{A} + \bar{L} \), we have that each of the unassigned customers requires no more than \( \bar{A}' \) units of capacity, so that it would suffice if an aggregate of \((|S| + |I|)\bar{A}'\) units of capacity among all facilities were available. Now let \( b' = \sum_{i \in \mathcal{I}} b'_i < (|S| + |I|)\bar{A}' \) denote the aggregate remaining capacity at the end of the greedy phase of the heuristic, and recall that, by Corollary 2, we know that \(|S| + |Q| \leq |I|\). Sufficient capacity can therefore be made available if, in the improvement phase, we are able to reduce to their lower bounds the level of \( \left[ \left( (|S| + |I|)\bar{A}' - b' \right) / \bar{D} \right] \) customers that the greedy phase of the heuristic assigned at their upper bounds. Since this number is independent of the number of customers, Lemma 4 implies that, for large enough \(|\mathcal{J}|\), this can indeed be done, implying that the heuristic is asymptotically feasible.

Finally, note that the objective function value of a feasible solution that is obtained by the heuristic satisfies

\[
Z^H \geq Z^{LP} - \left( \left[ \left( (|S| + |I|)\bar{A}' - b' \right) / \bar{D} \right] + |S| + |Q| \right) \bar{D} \bar{R} - |S|\left( \bar{P} - \bar{P} + (\bar{R} - \bar{R})\bar{L} \right).
\]

(5–19)

In the right hand side of this inequality, \((|S| + |I|)\bar{A}' - b' \) \( \bar{D} \bar{R} \) is the lost revenue from reducing customers to their lower bounds in order to assure enough aggregate capacity for unassigned customers in the greedy phase of the heuristic. The term \(|S|\left( \bar{P} - \bar{P} + (\bar{R} - \bar{R})\bar{L} \right)\) accounts
for the loss of fixed profit resulting from customers not assigned to the same facility as in (LP').

Clearly \( Z^{LP} \leq Z^*_{|J|} \leq Z^H \) implies

\[
\lim_{|J| \to \infty} \frac{1}{|J|} (Z^*_{|J|} - Z^H_{|J|}) \leq \lim_{|J| \to \infty} \frac{1}{|J|} (Z^{LP}_{|J|} - Z^H_{|J|}).
\]

Provided that the heuristic solution is feasible we then have

\[
\lim_{|J| \to \infty} \frac{1}{|J|} (Z^{LP}_{|J|} - Z^H_{|J|}) \\
\leq \lim_{|J| \to \infty} \frac{1}{|J|} \left( \left( \left( |S| + |I| \right) \bar{A}' - b' \right) D + |S| + |Q| \right) \bar{D} \bar{R} + |S| (\bar{P} - P + (\bar{R} - R) \bar{L})
\]
(by (5–19))

\[
\leq \lim_{|J| \to \infty} \frac{1}{|J|} \left( \left( 2 |I| \bar{A}' - b' \right) D + |I| \right) \bar{D} \bar{R} + |I| (\bar{P} - P + (\bar{R} - R) \bar{L})
\]
(by Corollary 2)

\[= 0.\]

Since the heuristic is asymptotically feasible, this implies that the heuristic is asymptotically optimal as well.

\[\square\]

5.2.2.2 Facility-dependent requirements

The result of Lemma 4 can unfortunately not be extended to the general case where customer requirements are facility-dependent, preventing us from extending the approach in the previous section to show asymptotic feasibility of the heuristic if we choose \( \lambda \) equal to an optimal dual vector \( \lambda^* \) of (LP'). In general, we therefore take a different approach: we choose \( \lambda \) equal to the optimal dual vector of an instance of (LP') in which the capacities have been reduced by an appropriately chosen small amount. Note, however, that we will still apply the two phases of the heuristic using the original capacities. By ensuring that the (temporary) capacity reductions are large enough to ensure that the customers that are fractionally assigned in the LP-relaxation can be assigned feasibly but,
at the same time, small enough for the corresponding solution to be close to optimal, we will be able to show that the greedy phase of the heuristic alone is asymptotically feasible and optimal.

More formally, consider an instance of the GAPFD with \(|\mathcal{J}|\) customers. Then associate with this instance a perturbed instance of (LP') in which all of the normalized capacities \(\beta_i (i \in \mathcal{I})\) are reduced by \(\delta_{|\mathcal{J}|}\), where

\[
\lim_{|\mathcal{J}| \to \infty} \delta_{|\mathcal{J}|} = 0 \quad (5-20)
\]

\[
\lim_{|\mathcal{J}| \to \infty} |\mathcal{J}| \delta_{|\mathcal{J}|} = \infty \quad (5-21)
\]

and

\[0 < \delta_{|\mathcal{J}|} \leq \delta < \Delta \leq \min_{i \in \mathcal{I}} \beta_i.\]

We will denote the perturbed problem by (LP'\((\delta_{|\mathcal{J}|})\)) and its optimal value by \(Z_{\mathcal{J}}^{LP'}(\delta_{|\mathcal{J}|})\).

The following preliminary result shows that the optimal values of the original and perturbed problems are very close.

**Lemma 5.** The optimal values of (LP') and (LP'\((\delta_{|\mathcal{J}|})\)) are close in the sense that, with probability one,

\[
\lim_{|\mathcal{J}| \to \infty} \frac{1}{|\mathcal{J}|} Z_{|\mathcal{J}|}^{LP'}(\delta_{|\mathcal{J}|}) = \lim_{|\mathcal{J}| \to \infty} \frac{1}{|\mathcal{J}|} Z_{|\mathcal{J}|}^{LP'}. \]

**Proof.** See Appendix A.

The following theorem employs this result to prove that we have a heuristic for the GAPFD that is asymptotically feasible and optimal.

**Theorem 8.** Consider problem instances generated according to our general stochastic model. Moreover, choose \(\lambda\) in the heuristic equal to an optimal dual vector to (LP'\((\delta_{|\mathcal{J}|})\)). Then the heuristic is asymptotically feasible and optimal.

**Proof.** As in the proof of Theorem 7, we may assume that the solution to the perturbed instance of (LP') is unique with respect to non-split customers, since we are only
interested in a probabilistic and asymptotic feasibility guarantee. Then Theorem 6 says that the greedy phase of the heuristic assigns no more than \(|S|\) customers to a different facility or to no facility at all, and no more than \(|Q|\) customers to the same facility but at a different level than the optimal solution to \((\text{LP}')\). Since, by Corollary 2, we know that \(|S| + |Q| \leq |I|\), it is easy to see that the additional amount of aggregate capacity over the amount used in the perturbed instance of \((\text{LP}')\) required for these customers is independent of \(|J|\). By (5–21) we can therefore conclude that, with probability one, the greedy phase of the heuristic yields a feasible solution to the GAPFD. Moreover, the objective function value of a feasible solution that is obtained by the greedy phase of the heuristic satisfies

\[
Z^G \geq Z^{'\text{LP}}(\delta_{|J|}) - |S| (\bar{R}(\bar{L} + \bar{D}) - RL) - |Q|(\bar{D}\bar{R}) - |S|(\bar{P} - P + (\bar{R} - R)\bar{L})
\] (5–22)

so that, in that case,

\[
\lim_{|J| \to \infty} \frac{1}{|J|} \left( Z^*_{|J|} - Z^H_{|J|} \right) \leq \lim_{|J| \to \infty} \frac{1}{|J|} \left( Z^*_{|J|} - Z^G_{|J|} \right)
\]

(since \(Z^G \leq Z^H\))

\[
\leq \lim_{|J| \to \infty} \frac{1}{|J|} \left( Z^{'\text{LP}}_{|J|} - Z^G_{|J|} \right)
\]

(since \(Z^* \leq Z^{'\text{LP}}\))

\[
= \lim_{|J| \to \infty} \frac{1}{|J|} \left( Z^{'\text{LP}}_{|J|}(\delta_{|J|}) - Z^G_{|J|} \right)
\]

(whith probability one as \(|J| \to \infty\), by Lemma 5)

\[
\leq \lim_{|J| \to \infty} \frac{1}{|J|} \left( |S| (\bar{R}(\bar{L} + \bar{D}) - RL) + |Q|(\bar{D}\bar{R}) + |S| (\bar{P} - P + (\bar{R} - R)\bar{L}) \right)
\]

(by (5–22))

\[
\leq \lim_{|J| \to \infty} \frac{1}{|J|} \left( (|S| + |Q|)(\bar{R}(\bar{L} + \bar{D}) - RL) + |S| (\bar{P} - P + (\bar{R} - R)\bar{L}) \right)
\]

(by Corollary 2)
\[
\lim_{|\mathcal{J}| \to \infty} \frac{|\mathcal{I}|}{|\mathcal{J}|} \left( \tilde{R}(\tilde{L} + \tilde{D}) - RL + \tilde{P} - P + (\tilde{R} - R)(\tilde{L}) \right)
\]
\[
= 0.
\]

Since the heuristic is asymptotically feasible, this implies that the heuristic is asymptotically optimal as well.

It is interesting to note that Theorem 8 actually shows that the greedy phase of the heuristic alone is asymptotically feasible and optimal.

### 5.2.3 Model Extension

It is interesting to note that our heuristic can still be applied (and retain the associated theoretical properties) if the variable revenue function is convex rather than linear, i.e., if the term \( r_{ij}v_{ij} \) in the objective function is replaced by \( \tilde{r}_{ij}(v_{ij}) \) where \( \tilde{r}_{ij} \) is a convex and nondecreasing function. Such a revenue function may be relevant from a practical point of view by realizing that a customer may be willing to pay an increasing amount per unit of product supplied within the acceptable range. In fact, in light of the discussion in Section 5.1, our model could accommodate a situation that exhibits both economies of scale for the supplier (through the fixed profit term) and a larger marginal value to customers who receive additional units of product. To apply our heuristics to this model generalization, we can simply linearize the revenue function by defining

\[
r_{ij} = \frac{(\tilde{r}_{ij}(u_{ij}) - \tilde{r}_{ij}(\ell_{ij}))/u_{ij} - \ell_{ij}).
\]

The asymptotic performance guarantees then follow in a relatively straightforward manner by realizing that the solution to the LP-relaxation overestimates the customer revenue for no more than \(|\mathcal{I}| \) customers.

### 5.3 Heuristic Improvement Issues

#### 5.3.1 Solution Improvement

Recall that our statement of the heuristic approach left some flexibility, in particular in Steps 5 and 7 of the improvement phase. Although a basic implementation was sufficient to obtain asymptotic performance guarantees, we will in this section discuss
a more sophisticated implementation which is guaranteed to yield superior results in practice.

5.3.1.1 Improvement phase

First, consider the selection of a set $A'$ of customers whose customer levels will be reduced to their lower bound in Step 5. Rather than identifying an arbitrary set that satisfies the properties specified in the heuristic, we sequentially add assignments from $A$ to $A'$ in the reverse of the order in which they were assigned in the greedy phase of the heuristic, until the set $A'$ satisfies the desired properties. Next, rather than attempting to arbitrarily assign customers in $\tilde{J}$ to facilities (in Step 7) we use the modified greedy algorithm proposed by Romeijn and Romero Morales [79] to solve the following instance of the GAP:

$$\text{maximize } \sum_{i \in I} \sum_{j \in J} p'_{ij}x_{ij}$$

subject to

$$\sum_{j \in J} a'_{ij}x_{ij} \leq b'_i \quad i \in I$$
$$\sum_{i \in I} x_{ij} = 1 \quad j \in \tilde{J}$$
$$x_{ij} \in \{0, 1\} \quad i \in I; j \in \tilde{J}$$

where $p'_{ij} = p_{ij} + r_{ij}\ell_{ij}$ and $a'_{ij} = a_{ij} + \ell_{ij} (i \in I; j \in J)$. Denote the optimal solution to (I) by $x^I$. The heuristic solution is updated by setting

$$x^H_{ij} = x^I_{ij} \text{ and } v^H_{ij} = x^I_{ij}\ell_{ij} \quad i \in I; j \in \tilde{J}. \quad (5–23)$$

5.3.1.2 Post-processing phase

Both the greedy phase of the heuristic in Section 5.2.1 and the improvement phase described in Section 5.3.1.1 are designed to provide high quality feasible solutions. However, it may still be possible to improve the quality of the solution. In particular,
given that a feasible assignment $x^H$ has been obtained (in either the greedy or improvement phase), we propose to optimally determine corresponding customer demand fulfillment levels $v^P$ (where the superscript P denotes the solution after the post-processing phase).

In fact, if $B_i = \{ j : x^H_{ij} = 1 \}$ is the set of customers assigned to facility $i$, the optimal customer demand fulfillment levels can be determined as follows. First, solve the following continuous knapsack problems for $i \in I$:

$$\begin{align*}
\text{maximize} & \quad \sum_{j \in B_i} r_{ij} w_{ij} \\
\text{subject to} & \quad \sum_{j \in B_i} w_{ij} \leq b_i - \sum_{j \in B_i} a_{ij} \\
& \quad 0 \leq w_{ij} \leq u_{ij} - \ell_{ij} \quad j \in B_i.
\end{align*}$$

Then, if $w^*$ denotes the optimal solutions to these problems, set $v^P_{ij} = w^*_{ij}$. It is easy to see that the problems $(KP_i)$ can be solved by choosing the flexible component of the sizes of the customers in $B_i$ as large as possible in nonincreasing order of $r_{ij}$ as long as facility capacity allows.

### 5.3.2 Capacity Perturbation Scheme

Recall that, for the general case where requirements are allowed to be facility-dependent, we reduce the normalized facility capacities $\beta_i$ by some amount $\delta_i |J|$ within the framework of Section 5.2.2.2. Despite the asymptotic feasibility result of Theorem 8, it is of course still possible that the heuristic fails to find a feasible solution. In particular, an inappropriate perturbation may lead to infeasibility for one of the following two reasons:

(i) If $\delta_i |J|$ is too large, the resulting perturbed capacities may be such that the instance of $(LP'(\delta_i |J|))$ is infeasible so that we cannot perform the greedy phase of the heuristic.

(ii) If $\delta_i |J|$ is too small, then we fail to reserve enough capacity to accommodate the customers that remain unassigned in the greedy phase.
We propose to use this information to iteratively modify the capacity perturbation as needed. Note that, to ensure that no perturbed facility capacities are nonpositive, we should initially have $0 \equiv \tilde{\delta} < \delta_{|\mathcal{J}|} < \bar{\delta} \equiv \min_{i \in \mathcal{I}} \beta_i$. In case the heuristic is unsuccessful due to (i), we update $\bar{\delta} = \delta_{|\mathcal{J}|}$ and decrease $\delta_{|\mathcal{J}|}$. If, on the other hand, it is unsuccessful due to (ii), we update $\tilde{\delta} = \delta_{|\mathcal{J}|}$ and increase $\delta_{|\mathcal{J}|}$. In either case, we set the new capacity perturbation to

$$
\delta_{|\mathcal{J}|} = \tilde{\delta} + \omega \times (\bar{\delta} - \tilde{\delta})
$$

(\text{where } \omega \in (0, 1)) and reapply the heuristic.

For instances with facility-independent requirements, no perturbation is required to obtain asymptotic performance guarantees. However, as for instances with facility-dependent requirements, it is of course possible that the heuristic does not find a feasible solution. In that case, we can apply the same iterative scheme, recognizing that we initially have $\delta_{|\mathcal{J}|} = 0$.

## 5.4 Computational Results

In this section we test the performance of our heuristics on a large set of randomly generated test problems. Following the theoretical results, we separately consider problems with facility-independent and facility-dependent requirements.

### 5.4.1 Experimental Design

We use the stochastic model given in Section 5.2.2 as the basis for generating problem instances. We consider instances with $|\mathcal{I}| = 15$ and $|\mathcal{I}| = 30$ facilities, and $|\mathcal{J}| = 5|\mathcal{I}|, 10|\mathcal{I}|, 25|\mathcal{I}|, 50|\mathcal{I}|$, and $100|\mathcal{I}|$ customers. For each customer, we generate the vectors of revenue parameters $R_j$ and $P_j$ independently from uniform distributions on $[1, 2]$ and $[30, 50]$, respectively. The customer requirements $A_j$, $L_j$ and $D_j$ are generated from uniform distributions on $[10, 20]$, $[75, 125]$, and $[15, 35]$, respectively. Note that, for instances with facility-independent requirements, only a single value for each of these parameters is generated, while for instances with facility-dependent requirements, we generate $|\mathcal{I}|$ values (one for each facility) independently from the specified distributions.
We focused on instances in which the facility-capacities were identical, that is, we set 
\[ b_i = \beta |J| \ (i \in I). \] 
For instances with facility-independent requirements, the value \( \Delta \) in Assumption 1 then reduces to

\[
\Delta = \beta - \frac{E(A_{ij} + L_{ij})}{|I|} \tag{5–24}
\]

and we therefore consider capacities of the form

\[
\beta = \tau \cdot \frac{E(A_{ij} + L_{ij})}{|I|}. \tag{5–25}
\]

For instances with facility-dependent requirements, Romeijn and Romero Morales [80] showed that the value \( \Delta \) in Assumption 1 reduces to

\[
\Delta = \beta - \frac{E(\min_{i \in I}(A_{i1} + L_{i1}))}{|I|} \tag{5–26}
\]

provided that the (lower bounds of the) customer requirements \( A_{i1} + L_{i1} \) are independent and have an increasing failure rate distribution, as is the case in our experiments. We therefore consider capacities of the form

\[
\beta = \tau \cdot \frac{E(\min_{i \in I}(A_{i1} + L_{i1}))}{|I|}. \tag{5–27}
\]

It is easy to see that in both (5–25) and (5–27), \( \tau > 1 \) is equivalent to Assumption 1 being satisfied. Moreover, these choices ensure that the tightness of the instances across different values of \( |I| \) is comparable for a given value of \( \tau \). In our experiments, we have considered values of \( \tau = 1.1, 1.2, \) and \( 1.3. \) The former two values correspond to cases where the capacity constraints are expected to have a strong limiting effect on the customer demand fulfillment levels that can be accommodated. The third value corresponds to loosely capacitated instances since they yield that the expressions in (5–24) and (5–26) are positive even when the ranges \( D_{i1} \) are added to the customer requirements.

We apply both the greedy phase and the improvement phase to each problem instance. We use the iterative capacity perturbation scheme described in Section 5.3.2.
to improve the ability of the heuristic to find feasible solutions for smaller values of $|J|$; here we simply set the update parameter equal to $\omega = \frac{1}{2}$. Moreover, unless otherwise noted, the post-processing phase described in Section 5.3.1.2 is applied as well. Finally, for a meaningful assessment of the heuristic performance we run CPLEX until a solution is obtained whose objective function value is at least as good as the one found by the heuristic or until 15 minutes of CPU time have been used. For each problem class, we present average results of 25 randomly generated instances. All experiments were performed on a PC with a 3.40 GHz Pentium IV processor and 2 GB of RAM, and all mixed-integer and linear programming problems were solved using CPLEX 10.1. The tables report

(i) the number of instances in which the heuristic found a feasible solution,

(ii) an upper bound on the relative solution error as measured by

$$\text{error} = \frac{z_{\text{LP}} - z_{\text{H}}}{z_{\text{H}}} \times 100\%,$$

(where, since the error is meaningless if no feasible solution is found, the average error is determined with respect to the instances for which the heuristic is able to find a feasible solution only),

(iii) the average CPU time used by the heuristic, for both the greedy phase and the greedy phase followed by the improvement phase over all iterations,

(iv) the average number of capacity perturbation iterations performed,

(v) the CPU time required by CPLEX, and

(vi) the number of instances for which CPLEX failed to obtain a solution of the desired quality within the allotted time (indicated by a superscript).

### 5.4.2 Facility-Independent Requirements

Tables 5-1–5-6 summarize the results obtained with our heuristics when applied to instances generated according to the model described in the previous section with facility-independent requirements. Recall that Theorem 7 says that the heuristic formed by the greedy and improvement phases is asymptotically feasible and optimal. However,
a similar guarantee cannot be given for the greedy phase alone; in particular, applying
the greedy phase alone does not guarantee asymptotic feasibility. The computational
results confirm this: for the two classes in which the capacity constraints are tightest
($\tau = 1.1$ and $\tau = 1.2$), the greedy phase is not able to find a feasible solution in any of
the instances generated. In contrast, when the greedy phase is considered in conjunction
with the improvement phase a feasible solution is obtained for almost all instances, with
the exception of instances with $|J| = 5|I|$ and $\tau = 1.1$ (for both $|I| = 15$ and $|I| = 30$).
However, our results show that, for such instances, performing the iterative capacity
perturbation scheme of Section 5.3.2 yields a feasible solution in all but a single instance
(with $|I| = 15$, $|J| = 75$, and $\tau = 1.1$). Note that, when $\tau = 1.3$, the greedy phase alone
is able to find a feasible solution for all instances. This can partly be explained by the fact
that, for large $|J|$, it can be expected that all customers can be performed at their upper
bound, making the improvement phase unnecessary even from a theoretical point of view.
It is noteworthy, however, that the greedy phase alone still performs very well for instances
with smaller values of $|J|$ when $\tau = 1.3$, despite the lack of any theoretical feasibility
guarantee.

The results clearly show that the average error approaches zero as the number of
customers increases. It is interesting to note that, for both values of $|I|$, the average errors
are largest when $\tau = 1.2$. This behavior is a consequence of the nature of the improvement
phase, in which the heuristic creates capacity for unassigned customers by decreasing
the size of already assigned customers to their lower bounds. When $\tau = 1.2$, customers
can generally be performed at higher levels than when $\tau = 1.1$, so that the net effect
of the improvement phase on solution quality is understandably larger for $\tau = 1.2$ than
for $\tau = 1.1$. However, this pattern does not continue to $\tau = 1.3$ since, as we concluded
above, the improvement phase is not required for these instances. We also remark that the
solution errors depend mainly on the ratio $|J|/|I|$ between the number of customers and
the number of facilities.
The heuristic is computationally very efficient, on average taking only slightly more than 1 second of CPU time when $|\mathcal{I}| = 15$ and about 4 seconds of CPU time when $|\mathcal{I}| = 30$. For smaller problem instances, CPLEX is able to find solutions of the same quality very rapidly as well. However, for larger instances and as the capacity increases, CPLEX is up to approximately 10 times slower for $|\mathcal{I}| = 15$ and up to approximately 50 times slower for $|\mathcal{I}| = 30$. Our heuristic is therefore especially promising for large instances and cases where the GAPFD needs to be solved repeatedly, for example under different scenarios or when it is a subproblem in a more complex strategic optimization problem. Perhaps surprisingly, despite the fact that an instance of (LP$'$) has to be solved, the time required by the heuristic increases only modestly (approximately linearly) in the size of the problem.

5.4.3 Facility-Dependent Requirements

Recall that, for instances with facility-dependent requirements, the heuristic employs the (dual) solution to the LP-relaxation of an instance of the GAPFD in which the normalized capacities $\beta$ are reduced by a quantity $\delta_{|\mathcal{J}|}$ satisfying (5–20) and (5–21) to ensure asymptotic feasibility and optimality. To ensure that no perturbed facility capacities are nonpositive for any value of $|\mathcal{J}|$, we propose to choose

$$\delta_{|\mathcal{J}|} = \alpha \times \frac{\beta}{\sqrt{|\mathcal{J}|}} \quad (5–28)$$

where $0 < \alpha < 1$, and where the magnitude of $\alpha$ represents a tradeoff between feasibility and solution quality. In our computational experiments we simply use $\alpha = \frac{1}{2}$.

Tables 5-7–5-12 summarize the results obtained with our heuristics when applied to instances generated according to the model described in Section 5.4.1 with facility-dependent requirements. Recall that Theorem 8 says that the greedy phase of the heuristic is asymptotically feasible and optimal. Although the greedy phase alone fails to find a feasible solution to a substantial number of problem instances for smaller ratios $|\mathcal{J}|/|\mathcal{I}|$, the greedy phase is uniformly successful for larger ratios. Moreover, the pattern of average
errors after the greedy phase alone shows (conditionally on finding a feasible solution) a
decreasing trend, illustrating the theoretical asymptotic optimality result.

The contribution to feasibility of the iterative capacity perturbation scheme is
apparent for instances with $|\mathcal{J}| \leq 25|\mathcal{I}|$: a feasible solution was obtained in all instances.
It should not be surprising that this results in an increase in average error, since the
instances for which a single iteration of the greedy phase is not able to find a feasible
solution are clearly the harder ones. However, the pattern of average errors still exhibits a
strongly decreasing trend, again clearly illustrating the theoretical asymptotic optimality
result.

The average CPU time required by the heuristic is substantially larger for instances
with facility-dependent requirements than for instances with facility-independent
requirements, particularly when $|\mathcal{I}| = 30$. CPLEX is able to solve instances with loose
capacities ($\tau = 1.3$) in about twice the time required by the heuristic (and even in only
about 50% more time for the smallest instances). The instances with tighter capacities
clearly illustrate the strength of the heuristic. When $\tau = 1.1$ and $|\mathcal{J}| \geq 25|\mathcal{I}|$, CPLEX
was not able to find a feasible solution within 15 minutes of CPU time for any instance,
and was successful for only 2 instances with $|\mathcal{J}| = 10|\mathcal{I}|$ and $|\mathcal{I}| = 30$. When $\tau = 1.2$
we see a similar behavior for $|\mathcal{J}| \geq 10|\mathcal{I}|$ and $|\mathcal{I}| = 30$. For $|\mathcal{I}| = 15$ and $|\mathcal{J}| = 10|\mathcal{I}|$
the time required by CPLEX exceeds that of the heuristic by a factor of 100, while for
$|\mathcal{J}| \geq 25|\mathcal{I}|$ CPLEX is again not successful in a substantial number of problem instances.
(Note that the computation times for CPLEX are averaged over only those instances in
which it was successful and therefore do not include the instances for which CPLEX was
unsuccessful within 15 minutes. Moreover, for each instance in which the CPU time limit
expired, CPLEX had not yet found a feasible solution.)

5.4.4 Effect of Post-Processing Phase

The asymptotic feasibility and optimality guarantees of the heuristic for instances
with facility-independent as well as instances with facility-dependent requirements hold
even in the absence of the post-processing procedure described in Section 5.3.1.2. However, as mentioned above, the results presented thus far pertain to solutions that have been improved by this post-processing phase. Therefore, we will in this section study the effect of the post-processing phase on the quality of the solutions, illustrating simultaneously the asymptotic performance guarantees without the post-processing phase as well as the practical importance of applying this phase.

Tables 5-13 and 5-14 summarize these results. For brevity, we have only focused on instances with intermediate capacities ($\tau = 1.2$); however, the results for other values of $\tau$ are qualitatively similar. For the case of facility-independent requirements we have omitted the results of the greedy phase alone, since this phase was never able to find a feasible solution and thus the post-processing phase is irrelevant. In both tables, the columns labeled “before” contain the solution errors without post-processing, while the columns labeled “after” contain the solution errors with post-processing. We see that, in all cases and for both types of problem instances, the results without post-processing phase are consistent with the asymptotic optimality guarantees. However, we also see that the post-processing phase substantially reduces solution error, particularly for problem instances with a small ratio of $|\mathcal{J}|/|\mathcal{I}|$.

### 5.5 Conclusions

This chapter considered the GAPFD, which generalizes the classical GAP. Our extension applies to situations in which, along with the assignment of customers to facilities, a flexible degree of resource consumption must be determined for each of these assignments. To solve the GAPFD, we propose a class of heuristics motivated by attractive properties of the optimal solution of the LP-relaxation to the GAPFD and its corresponding dual. For two classes of customer requirements we show that an implementation of the heuristic exists that is asymptotically feasible and optimal with probability one under a very broad stochastic model on the problem parameters. Our computational study demonstrates that the heuristic performs very well, particularly
for large ratios of the number of customers to the number of facilities. When additional improvement strategies that we propose in this chapter are also considered, the heuristic is successful on instances with smaller ratios as well. We observe that the time required to obtain solutions of comparable quality is considerably less for our heuristic than for the commercial solver CPLEX. The fact that our heuristic obtains quality solutions so quickly is encouraging for further research directions. Specifically, we believe that the heuristic may be very valuable when solving more general related optimization problems for which the GAPFD arises as a subproblem that needs to be solved repeatedly.
Table 5-1. Facility-independent requirements: 15 facilities, $\tau = 1.1$

| $|\mathcal{J}|$ | Greedy phase | Improvement phase | CPLEX |
|---|---|---|---|
| | feas. error (%) | time (sec.) | feas. error (%) | time (sec.) | it. |
| 75 | 0 | 0.04 | 24 | 1.12 | 0.33 | 5.84 |
| 150 | 0 | 0.09 | 25 | 0.38 | 0.13 | 1 |
| 375 | 0 | 0.20 | 25 | 0.09 | 0.29 | 1 |
| 750 | 0 | 0.35 | 25 | 0.05 | 0.51 | 1 |
| 1,500 | 0 | 0.72 | 25 | 0.03 | 1.03 | 1 |

Table 5-2. Facility-independent requirements: 15 facilities, $\tau = 1.2$

| $|\mathcal{J}|$ | Greedy phase | Improvement phase | CPLEX |
|---|---|---|---|
| | feas. error (%) | time (sec.) | feas. error (%) | time (sec.) | it. |
| 75 | 0 | 0.04 | 25 | 3.20 | 0.07 | 1 |
| 150 | 0 | 0.07 | 25 | 1.82 | 0.11 | 1 |
| 375 | 0 | 0.18 | 25 | 0.47 | 0.27 | 1 |
| 750 | 0 | 0.33 | 25 | 0.11 | 0.49 | 1 |
| 1,500 | 0 | 0.68 | 25 | 0.02 | 0.98 | 1 |

Table 5-3. Facility-independent requirements: 15 facilities $\tau = 1.3$

| $|\mathcal{J}|$ | Greedy phase | Improvement phase | CPLEX |
|---|---|---|---|
| | feas. error (%) | time (sec.) | feas. error (%) | time (sec.) | it. |
| 75 | 25 | 0.79 | 0.06 | 25 | 0.79 | 0.06 | 1 |
| 150 | 25 | 0.24 | 0.10 | 25 | 0.24 | 0.10 | 1 |
| 375 | 25 | 0.06 | 0.24 | 25 | 0.06 | 0.24 | 1 |
| 750 | 25 | 0.02 | 0.45 | 25 | 0.02 | 0.45 | 1 |
| 1,500 | 25 | 0.00 | 0.91 | 25 | 0.00 | 0.91 | 1 |

Table 5-4. Facility-independent requirements: 30 facilities, $\tau = 1.1$

| $|\mathcal{J}|$ | Greedy phase | Improvement phase | CPLEX |
|---|---|---|---|
| | feas. error (%) | time (sec.) | feas. error (%) | time (sec.) | it. |
| 150 | 0 | 0.18 | 25 | 1.05 | 0.35 | 1.56 |
| 300 | 0 | 0.27 | 25 | 0.36 | 0.39 | 1 |
| 750 | 0 | 0.68 | 25 | 0.09 | 0.95 | 1 |
| 1,500 | 0 | 1.40 | 25 | 0.04 | 1.90 | 1 |
| 3,000 | 0 | 2.95 | 25 | 0.02 | 3.95 | 1 |

Table 5-5. Facility-independent requirements: 30 facilities, $\tau = 1.2$

| $|\mathcal{J}|$ | Greedy phase | Improvement phase | CPLEX |
|---|---|---|---|
| | feas. error (%) | time (sec.) | feas. error (%) | time (sec.) | it. |
| 150 | 0 | 0.15 | 25 | 3.56 | 0.21 | 1 |
| 300 | 0 | 0.26 | 25 | 1.60 | 0.38 | 1 |
| 750 | 0 | 0.66 | 25 | 0.38 | 0.92 | 1 |
| 1,500 | 0 | 1.34 | 25 | 0.12 | 1.83 | 1 |
| 3,000 | 0 | 2.73 | 25 | 0.02 | 3.71 | 1 |

105
Table 5-6. Facility-independent requirements: 30 facilities, $\tau = 1.3$

| $|\mathcal{J}|$ | Greedy phase | Improvement phase | CPLEX |
|---|---|---|---|
| feas. | error (%) | time (sec.) | feas. | error (%) | time (sec.) | it. | time (sec.) |
| 150 | 25 | 0.67 | 0.18 | 25 | 0.67 | 0.18 | 1 | 8.21 |
| 300 | 25 | 0.29 | 0.34 | 25 | 0.29 | 0.34 | 1 | 17.62 |
| 750 | 25 | 0.04 | 0.84 | 25 | 0.04 | 0.84 | 1 | 28.24 |
| 1,500 | 25 | 0.01 | 1.63 | 25 | 0.01 | 1.63 | 1 | 50.19 |
| 3,000 | 25 | 0.00 | 3.25 | 25 | 0.00 | 3.25 | 1 | 163.96 |

Table 5-7. Facility-dependent requirements: 15 facilities, $\tau = 1.1$

| $|\mathcal{J}|$ | Greedy phase | Improvement phase | CPLEX |
|---|---|---|---|
| feas. | error (%) | time (sec.) | feas. | error (%) | time (sec.) | it. | time (sec.) |
| 75 | 10 | 5.06 | 0.08 | 25 | 9.32 | 0.39 | 4.84 | 3.04 |
| 150 | 6 | 2.65 | 0.18 | 25 | 9.32 | 0.92 | 4.88 | 144.55(3) |
| 375 | 25 | 1.28 | 0.36 | 25 | 1.28 | 0.36 | 1 | — (25) |
| 750 | 25 | 0.80 | 0.78 | 25 | 0.80 | 0.78 | 1 | — (25) |
| 1,500 | 25 | 0.52 | 1.89 | 25 | 0.52 | 1.89 | 1 | — (25) |

Table 5-8. Facility-dependent requirements: 15 facilities, $\tau = 1.2$

| $|\mathcal{J}|$ | Greedy phase | Improvement phase | CPLEX |
|---|---|---|---|
| feas. | error (%) | time (sec.) | feas. | error (%) | time (sec.) | it. | time (sec.) |
| 75 | 4 | 1.55 | 0.07 | 25 | 7.17 | 0.34 | 4.36 | 1.56 |
| 150 | 15 | 0.61 | 0.17 | 25 | 2.68 | 0.48 | 2.60 | 50.30 |
| 375 | 25 | 0.21 | 0.31 | 25 | 0.21 | 0.31 | 1 | 341.90(10) |
| 750 | 25 | 0.10 | 0.65 | 25 | 0.10 | 0.65 | 1 | 478.38(20) |
| 1,500 | 25 | 0.05 | 1.41 | 25 | 0.05 | 1.41 | 1 | 333.57(23) |

Table 5-9. Facility-dependent requirements: 15 facilities, $\tau = 1.3$

| $|\mathcal{J}|$ | Greedy phase | Improvement phase | CPLEX |
|---|---|---|---|
| feas. | error (%) | time (sec.) | feas. | error (%) | time (sec.) | it. | time (sec.) |
| 75 | 9 | 0.97 | 0.07 | 25 | 8.90 | 0.22 | 2.68 | 0.38 |
| 150 | 19 | 0.29 | 0.16 | 25 | 0.66 | 0.20 | 1.12 | 0.47 |
| 375 | 25 | 0.02 | 0.29 | 25 | 0.02 | 0.29 | 1 | 0.81 |
| 750 | 25 | 0.01 | 0.57 | 25 | 0.01 | 0.57 | 1 | 1.72 |
| 1,500 | 25 | 0.00 | 1.17 | 25 | 0.00 | 1.17 | 1 | 4.50 |

Table 5-10. Facility-dependent requirements: 30 facilities, $\tau = 1.1$

| $|\mathcal{J}|$ | Greedy phase | Improvement phase | CPLEX |
|---|---|---|---|
| feas. | error (%) | time (sec.) | feas. | error (%) | time (sec.) | it. | time (sec.) |
| 150 | 1 | 2.25 | 0.40 | 25 | 13.38 | 3.10 | 7.72 | 139.74 |
| 300 | 0 | — | 0.50 | 25 | 8.03 | 5.37 | 7.00 | 440.51(23) |
| 750 | 14 | 0.79 | 1.76 | 25 | 2.60 | 10.83 | 3.64 | — (25) |
| 1,500 | 25 | 0.48 | 4.86 | 25 | 0.48 | 4.86 | 1 | — (25) |
| 3,000 | 25 | 0.31 | 14.11 | 25 | 0.31 | 14.11 | 1 | — (25) |
Table 5-11. Facility-dependent requirements: 30 facilities, $\tau = 1.2$

| $|J|$  | feas. error (%) | time (sec.) | feas. error (%) | time (sec.) | it. | CPLEX time (sec.) |
|------|-----------------|-------------|-----------------|-------------|-----|------------------|
| 150  | 0               | —           | 0.34            | 25          | 5.01| 2.21             |
| 300  | 0               | —           | 0.42            | 25          | 3.95| 4.21             |
| 750  | 20              | 0.22        | 1.44            | 25          | 0.67| 4.68             |
| 1,500| 25              | 0.09        | 3.54            | 25          | 0.09| 3.54             |
| 3,000| 25              | 0.06        | 8.93            | 25          | 0.06| 8.93             |

Table 5-12. Facility-dependent requirements: 30 facilities, $\tau = 1.3$

| $|J|$  | feas. error (%) | time (sec.) | feas. error (%) | time (sec.) | it. | CPLEX time (sec.) |
|------|-----------------|-------------|-----------------|-------------|-----|------------------|
| 150  | 0               | —           | 0.32            | 25          | 11.48| 1.69             |
| 300  | 4               | 0.35        | 0.38            | 25          | 1.91 | 1.15             |
| 750  | 21              | 0.04        | 1.20            | 25          | 0.04 | 1.25             |
| 1,500| 25              | 0.01        | 2.75            | 25          | 0.01 | 2.75             |
| 3,000| 25              | 0.00        | 6.38            | 25          | 0.00 | 6.38             |

Table 5-13. Post-processing effect on heuristic with greedy and improvement phase; facility-independent requirements: $\tau = 1.2$

| $|J|$  | error (%) | error (%) | error (%) | error (%) |
|------|-----------|-----------|-----------|-----------|
| 5$|I| 19.42 | 3.20 | 19.47 | 3.56 |
| 10$|I| 10.62 | 1.82 | 10.90 | 1.60 |
| 25$|I| 4.05 | 0.47 | 4.15 | 0.38 |
| 50$|I| 2.00 | 0.11 | 2.03 | 0.12 |
| 100$|I| 0.98 | 0.02 | 1.00 | 0.02 |

Table 5-14. Post-processing effect; facility-dependent requirements: $\tau = 1.2$

| $|J|$  | $|I| = 15$ | $|I| = 30$ |
|------|-----------|-----------|
| Greedy phase | Improvement phase | Greedy phase | Improvement phase |
| error (%) | error (%) | error (%) | error (%) |
| 5$|I| 7.92 | 1.55 | 21.30 | 7.17 |
| 10$|I| 4.91 | 0.61 | 10.48 | 2.68 |
| 25$|I| 2.78 | 0.21 | 2.78 | 0.21 |
| 50$|I| 1.91 | 0.10 | 1.91 | 0.10 |
| 100$|I| 1.32 | 0.05 | 1.32 | 0.05 |
In this chapter we propose heuristic procedures for a generalization of the well-known Capacitated Facility Location Problem with Single-Sourcing constraints (CFLP), known as the Capacitated Facility Location Problem with Flexible Demand (CFLFD). In addition to the standard problem scope of the CFLP, the CFLFD permits flexible customer demand specifications. That is, for a measurable product characteristic (e.g., weight, length, volume, units delivered), a customer specifies an allowable range for demand fulfillment. This assignment-based optimization problem falls into the class of challenging mixed-integer programs that become very difficult to solve as the number of customers per facility increases. Because many practical applications of this problem class require obtaining/updating solutions very quickly, it is important to identify fast heuristic solution methods that, on average, provide near-optimal solutions.

As mentioned in Chapter 2, the CFLP is a special-case of the CFLFD, our problem clearly belongs to the class of \( \mathcal{NP} \)-Hard optimization problems. The difficulty of such problems requires considering both exact and heuristic solution methodologies. Recent theoretical advances in integer programming have resulted in exact solution methodologies that have proven successful on previously unsolved problem instances. For example, the decomposition-based separation algorithm for the Capacitated Vehicle Routing Problem (CVRP), proposed by Ralphs et al. [76], solves three of the previously unsolved VRP instances from the TSPLIB repository presented by Reinelt [77]. Similarly, a stabilized branch-and-cut-and-price algorithm for the Generalized Assignment Problem (GAP), introduced by Pigatti et al. [74], was able to solve three previously unsolved instances from the OR-Library. Unfortunately, even with these innovative techniques, many real-world size problems can not be solved within practitioners’ time requirements. In fact, for numerous integer programming models that consider the assignment of customers to resources (i.e., the GAP, Savelsbergh [88] and Pigatti et al. [74]; the CVRP, Fukasaw
and the Multiperiod Single-Sourcing Problem (MPSSP), Freling et al. [37]), exact procedures are often successful only on instances with a small ratio of customers to resources, limiting the number of problems that can be solved within acceptable tolerance levels. Moreover, the number of scenarios for which decision makers must repeatedly revise their strategic plans is growing rapidly as information becomes available in real-time. For example, production planners now have access to changes in inventory levels, demand rates and resource levels, as they occur. To make use of this information, a planning schedule may require updates numerous times a day. In such cases, it is necessary to design an efficient heuristic to serve as either a supplement to an exact algorithm or as a stand-alone procedure that provides quality solutions with limited computational effort.

In Chapter 4 we proposed an exact algorithm for the CFLFD. As is common when applying branch-and-price to assignment problems, the success of the approach was limited to problems with a relatively small ratio of customers to facilities (usually up to about 10). In this chapter, we propose the first heuristic methodology targeted at solving the broad class of problem instances with a large ratio of the number of customers to the number of facilities. The heuristic approach we propose employs a combined facility neighborhood search method and a fast heuristic solution method for solving a generalization of the GAP. We discuss specific implementation issues related to this methodology, including methods for obtaining initial feasible solutions, effective ways to search a large neighborhood of solutions and efficient ways to develop hybrid approaches that combine successful individual heuristic methodologies. While successful heuristics have been developed for relevant problems, such as Ahuja et al.’s [6] multi-exchange heuristic CFLP, the notion of flexible demand provides additional challenges that are met through the approach proposed in this work. Computational tests illustrate the benefits of our proposed approach for solving problems in this class.

Balakrishnan and Geunes [13] proposed Lagrangian-based, bin-packing-based, and LP-rounding heuristics for the closely related Flexible Demand Assignment Problem
(FDA). However, their computational testing focused on instances with a small ratio of customers to facilities. As will be evident in the following sections, a reassignment of even a single customer in the CFLFD requires a subsequent assessment of the amount of demand fulfilled for each customer assigned to that particular facility. Therefore, the reassignment of customers in our approach will differ distinctly from that of previous work. In the case of the approach in Ahuja et al. [6], if the limited reassignment fails to find an improving or feasible solution, the authors consider a complete reassignment of customers, determined by heuristically solving an instance of the generalized assignment problem. It is important to emphasize that the method established in Ahuja et al. [6] allows for total customer reassignment only as a last resort. However, an important aspect of our approach is motivated, in part, by this less frequently employed complete reassignment step. We contend that with the availability of a quick, effective heuristic to solve the necessary assignment subproblem (and in light of the increased computational effort required to fully assess the partial reassignment of customers), the effort spent considering additional large neighborhoods (such as those proposed in Ahuja et al. [4]) is not necessary for a successful heuristic approach for the CFLFD. Moreover, since certain very large neighborhoods considered in previous work grow quadratically in the number of customers, these methods are not applicable for solving larger problem instances. The work of Ahuja et al. [6], considers instances with a facility to customer ratio of no more than 10. Therefore, the approach offered in this work serves as the first heuristic to consider a facility location problem with demand flexibility and offers a general solution framework that is particularly applicable to large-scale decomposable assignment problems that have received little attention in the literature.

The remainder of this paper is organized as follows. Section 6.1 formally introduces the version of the CFLFD studied in this chapter. Section 6.2 proposes two separate search techniques for solving the CFLFD. Implementation details of the heuristic approaches are presented in Section 6.3. In Section 6.4, we perform a computational
study of the various implementations of our heuristic on a broad collection of experiments. Finally, in Section 6.5, we discuss the application of our heuristic framework to similar fixed-charge problems and offer some concluding remarks.

6.1 Optimization and Model Formulation

The optimization model discussed in this chapter is the CFLFD presented in Chapter 3 with linear revenue functions (i.e. \( r_{ij}(v_{ij}) = r_{ij}(v_{ij}) \)). Furthermore, as in Chapter 5 we let \( J_i = \mathcal{J} \ (i \in \mathcal{I}) \). Specifically, the model considered in this section is given by problem as follows:

\[
\begin{align*}
\text{maximize} & \quad \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} r_{ij} v_{ij} + \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} p_{ij} x_{ij} - \sum_{i \in \mathcal{I}} f_i y_i \\
\text{subject to} & \quad (\text{CFLFD-L}) \\
& \quad \sum_{j \in \mathcal{J}} (a_{ij} x_{ij} + v_{ij}) \leq b_i y_i \quad i \in \mathcal{I} \quad (6-1) \\
& \quad \sum_{i \in \mathcal{I} : j \in \mathcal{J}} x_{ij} = 1 \quad j \in \mathcal{J} \quad (6-2) \\
& \quad v_{ij} \geq l_{ij} x_{ij} \quad i \in \mathcal{I}; \ j \in \mathcal{J} \quad (6-3) \\
& \quad v_{ij} \leq u_{ij} x_{ij} \quad i \in \mathcal{I}; \ j \in \mathcal{J} \quad (6-4) \\
& \quad x_{ij} \in \{0, 1\} \quad i \in \mathcal{I} ; \ j \in \mathcal{J} \quad (6-5) \\
& \quad y_i \in \{0, 1\} \quad i \in \mathcal{I} . \quad (6-6)
\end{align*}
\]

For the notational simplicity we will refer to CFLFD-L as CFLFD throughout this chapter, noting however that the most general version of the CFLFD is given in Chapter 3. In the next section, we propose a search heuristic that exploits the structure of the CFLFD in a way that enables us to consider a very large neighborhood.

6.2 Heuristic Framework

This section describes the components of a general search heuristic framework for the CFLFD. We consider techniques for exploring two classes of search neighborhoods. Section 6.2.1 describes the core of our heuristic. This approach searches a neighborhood defined by manipulating the set of open facilities. In this search procedure, all customers
are reassigned at each iteration. While a highly efficient methodology is used to determine the reassignment of all customers among a set of open facilities, the number of open facility sets to consider is still very large. In Section 6.2.2, we introduce a specialized implementation of a so-called very-large-neighborhood-search (VLSN) heuristic. This technique allows for multiple customer exchanges in any move; however, no more than one customer among those assigned to a particular facility may be reassigned in a single move. This secondary procedure is considered in an additional hybrid implementation discussed in Section 6.3.2.

6.2.1 Facility Neighborhood Search

Our primary search approach is designed to easily allow for improving solutions that correspond to opening or closing a facility. While the VLSN procedure described in Section 6.2.2 is effective in identifying alternative customer assignments among a given set of open facilities, it rarely identifies solutions that alter the set of open facilities. This limitation is confirmed by Ahuja et al.’s [6] inclusion of the facility neighborhood structure in their approach for CFLP. In this neighborhood, the opening, closing, or transferring of facilities is considered. To estimate the impact of each of these moves, they attempted to identify a subset of customers whose reassignment to a single different facility resulted in a cost savings. As mentioned in the previous section, the identification of potentially improving moves in Ahuja et al.’s [6] approach required the direct comparison of assignment cost parameters. However, in the case of the CFLFD, determining the change in profit from the reassignment of even a subset of customers requires that the level that demand is satisfied must be reassessed for each customer assigned to any facility which adds or loses a customer in the reassignment. Therefore, since considering reassignments of small subsets of customers is already more computationally intensive in the case of the CFLFD than CFLP, we focus immediately on determining a complete reassignment of all customers via an efficient heuristic when considering the various options with respect to opening and closing facilities. To contrast this procedure with
previous work, Ahuja et al.'s [6] approach only considered total reassignment of customers when their partial reassignment procedure was unable to find any improving feasible solution. Our strategy effectively allows us to consider single facility moves (open, close, or swap) and multi-exchange (multiple customers being reassigned among numerous facilities) in a single search step. Clearly this heuristic searches a very large neighborhood, and if implemented with a successful subproblem methodology, is likely to offer high quality solutions. Of course, reassigning all customers at each step of our search can be computationally intensive, as well. To compensate for this extra effort, we contend that, for problems with a large ratio of customers to facilities, relying on an efficiently obtained subproblem solution to determine customer assignments is at least as effective as VLSN. Alternatively, we focus on an intelligent implementation that efficiently searches this very large facility neighborhood and determines a set of open facilities and subsequent customer assignments which correspond to a high quality solution. In the remainder of this section we present the search framework and introduce the relevant subproblem to be solved in the secondary stage of our search.

Our framework allows for a search of a very large neighborhood. In each potential move, our search evaluates the benefit of manipulating the set of open facilities and reassigning customers among these facilities. Specifically, for any partial feasible solution to (CFLFD), \((x^N, v^N, y^N)\), let \(O\) be the set of open facilities in this solution, i.e.,

\[ O = \{i \in I : y_i^N = 1\} \]

and \(C\) be the set of closed facilities, i.e.,

\[ C = \{i \in I : y_i^N = 0\}. \]

Our heuristic can be described by three separate moves.

- **Close**: close a single facility in \(O\) and reassign all customers;
- **Open**: open a single facility in \(C\) and reassign all customers;
• Swap: close a single facility in $O$ and replace it by opening a closed facility in $C$, and then reassign all customers.

For each neighborhood move described above, reassigning all customers among a given set of open facilities, $\tilde{I}$, and determining the corresponding demand fulfillment levels, is accomplished by solving the following mixed-integer program

\[
\text{maximize } \sum_{i \in \tilde{I}} \sum_{j \in J} p_{ij}x_{ij} + \sum_{i \in \tilde{I}} \sum_{j \in J} r_{ij}v_{ij}
\]

subject to

\[
\sum_{j \in J} (a_{ij}x_{ij} + v_{ij}) \leq b_i \quad i \in \tilde{I} \tag{6-7}
\]

\[
\sum_{i \in \tilde{I}} x_{ij} = 1 \quad j \in J \tag{6-8}
\]

\[
\ell_{ij}x_{ij} \leq v_{ij} \leq u_{ij}x_{ij} \quad i \in \tilde{I}, j \in J \tag{6-9}
\]

\[
x_{ij} \in \{0, 1\} \quad i \in \tilde{I}, j \in J. \tag{6-10}
\]

This problem is the Generalized Assignment Problem with Flexible Jobs (with linear revenue functions), which was studied in Chapter 5. This class of optimization problems is clearly $NP$-Hard since the GAP is an important special case. However, in the previous chapter we developed an efficient constructive heuristic for the GAPFD, which was shown to be asymptotically feasible and optimal under a very general stochastic model.

This framework can clearly be implemented in a variety of ways. To produce an efficient implementation, important design decisions must be made. Section 6.3 discusses each of these issues and presents the most successful implementation determined through our computational study. However, before considering these implementation issues we introduce an additional large-scale neighborhood search technique that has been applied to numerous set-partitioning optimization problems.
6.2.2 Single-Customer VLSN

In this section we discuss a unique VLSN implementation for the CFLFD which we will consider as a post-processing phase for our main facility neighborhood search procedure. In recent literature, a variety of assignment-based optimization problems have been solved using VLSN, e.g., the Multi-Period Single-Sourcing Problem (Ahuja et al. [3]), the Single Source Capacitated Facility Location Problem (Ahuja et al. [6]), and Vehicle Routing and Scheduling Problems (Thompson and Psaraftis [97]). Details of the VLSN procedure are well documented. Therefore, this section only offers details of considerations made to accommodate the unique demand flexibility component of our problem. For a detailed survey of the VLSN technique, the reader is referred to Ahuja et al. [5]. For a specific discussion of single-customer multi-exchange VLSN applied to the CFLP, a comprehensive discussion is provided in Ahuja et al. [6, Section 4].

In general, the single-customer multi-exchange search neighborhood is explored by constructing a so-called improvement graph. The improvement graph consists of a node for each customer, separate nodes for each facility, and an origin node. Arcs connect each pair of ‘customer nodes’, provided that the customers are assigned to different facilities. In addition, arcs connect each ‘customer node’ to each ‘facility node’, excluding the node representing the facility to which the customer is currently assigned. Lastly, the graph includes arcs from each ‘customer node’ to the origin node, as well as an arc from each ‘facility node’ back to the origin. The inclusion of ‘facility nodes’ and the origin nodes allows for exchanges in which a customer is added to (removed from) a facility, but no customer is relinquished from (added to) that facility. Using this representation, an improving move is obtained by identifying a negative subset-disjoint cycle in the network. A comprehensive discussion of disjoint cycles and the optimization effort required to identify them is found in Thompson and Orlin [96]. For the purposes of this work, we apply the effective heuristic proposed by Ahuja et al. [7] to the \(\mathcal{NP}\)-Hard problem of identifying a negative subset-disjoint cycle.
Thus far, the implementation of VLSN on our problem is no different than when applied to any similar problem with a set-partitioning structure. However, unlike any model previously solved with VLSN, the flexible demand component of the CFLFD must be accounted for when determining the cost of the arcs contained in the improvement graph. In general, the cost of any arc in the improvement graph is simply the difference in total profit resulting from assignment change(s) represented by that arc. In the CFLP, as well as other models with fixed assignment profits or costs, this difference is dependent solely on the cost (profit) parameters in the problem. In the case of the CFLFD, each assignment exchange must be accompanied by a corresponding demand fulfillment level decision. Therefore, to calculate the cost of an arc, a subproblem must be solved to determine the appropriate change in customer demand fulfillment levels associated with each exchange.

To illustrate, consider two ‘customer nodes’, $j_1$ and $j_2$. Let $i_{j_2}$ be the facility to which customer $j_2$ is currently assigned and $\bar{J}$ the set of customers currently assigned to $i_{j_2}$ The cost of the arc connecting ‘customer nodes’, $j_1$ and $j_2$ is

$$z_{\bar{J}}^{SP} - z_{\{\bar{J}, j_2\} \cup \{j_1\}}^{SP},$$

where, for any set of customers $\bar{J}$ assigned to $\bar{i}$, the value $z_{\bar{J}}^{SP}$ is obtained by solving the following optimization problem, (SP),

$$\theta + \maximize \sum_{j \in \mathcal{J}} r_{ij} w_{ij}$$

subject to

$$\sum_{j \in \mathcal{J}} w_{ij} \leq b_i - \sum_{j \in \mathcal{J}} (a_{ij} + \ell_{ij}),$$

$$0 \leq w_{ij} \leq u_{ij} - \ell_{ij} \quad j \in \bar{J},$$

116
and $\theta$ is a constant equal to $\sum_{j \in \bar{J}} (p_{ij} + r_{ij} \ell_{ij}) - f_i$. In this formulation, the decision variable $w_{ij}$ represents the total amount of resource (i.e., both fixed and variable) consumed by customer $j$ (when assigned to facility $\bar{i}$). It is easy to see that SP can be treated in the same manner as a 0/1 continuous knapsack problem. The problem can be solved by choosing the flexible component of the sizes of the demand fulfillment as large as possible in nonincreasing order of $r_{ij}$ as long as capacity allows. In a similar manner, SP can be used to determine the appropriate costs of arcs connecting the remaining nodes.

The ability to efficiently calculate true arc costs is a notable result that is unique to the development of a heuristic for the CFLFD. With the availability of a procedure to solve SP efficiently to optimality, we avoid the reduced impact of VLSN that is generally caused by having to rely on arc cost estimates. Thus, even with the additional demand flexibility decision component, a large neighborhood can be efficiently explored for problem sizes similar to those considered in related work. While the literature has shown that this technique is effective in determining customer assignments for a fixed set of facilities, establishing an appropriate set of open facilities is a weakness of the implementation described in this section. Therefore, this approach will be utilized only in the limited manner outlined at the end of the following section.

6.3 Search Heuristic Implementation

This section will primarily focus on implementation choices regarding the neighborhood search heuristic proposed in Section 6.2.1. However, at the end of the section, the motivation for a hybrid approach which melds the facility neighborhood search (FS) approach with the VLSN in Section 6.2.2 is presented. With regard to FS, a number of key considerations are necessary. First, since FS falls in the class of improvement heuristics, we must determine how an initial feasible solution is obtained. Second, the effort required to consider a full set of facility moves is extensive. Therefore, consideration should be given to intelligently consider a subset of potentially attractive moves. Then, of course, the criteria used to determine search termination must be specified. The choices
made with respect to each of these issues has a considerable effect on the success of our heuristic. In this section we treat each of these issues separately. We offer alternatives for each issue, discuss our findings with respect to each option, and provide the best implementation encountered.

6.3.1 Initial Feasible Solution

Since alternative heuristics for the CFLFD do not exist that may serve as a source for an initial feasible solution, we study two alternatives to determine a starting solution. The first is motivated by the availability of an efficient heuristic to solve the GAPFD. In this alternative we assume that all facilities are open, and then solve a GAPFD via the constructive heuristic proposed in Chapter 5 to determine the customer assignments and the level at which each customer’s demand is satisfied. This alternative has clear advantages. First, with the capacity of all facilities available for customer assignments, it is relatively easy to find a feasible solution to the corresponding GAPFD. The results in Chapter 5 indicate that solutions obtained by their GAPFD heuristic for instances with large amounts of available facility capacity are very close to optimal for a fixed set of facilities, and can be determined with a minimal amount of computational effort. Unfortunately, the disadvantage of this alternative is that the facility procurement costs are neglected. Therefore, while easy to implement and intuitive to consider, the quality of the solutions obtained using this approach were of poor quality and ultimately resulted in prolonged duration of our search heuristic.

The second alternative is a modified random solution generation approach. This method for generating an initial feasible solution is commonly used to start an improvement heuristic search (e.g., the Traveling Salesman Problem, Lin and Kernighan [61]; and the Resource Constrained Project Scheduling Problem, Lee and Kim [59]). When randomly generating a feasible solution to the CFLFD, we explicitly attempt to minimize the number of facilities which are ‘opened’ in the solution. Let \( \mathcal{J} \) and \( \mathcal{I} \) be the set of of unassigned customers and unused facilities, respectively. Furthermore, let \( b_i' \) be the
remaining capacity of facility $i$ ($i \in I$). The procedure for randomly generating a feasible solution to the CFLFD is as follows.

**Randomly generate CFLFD solution**

**Step 0.** Set $\bar{J} = J$, $\bar{I} = I$ and $b'_i = b_i$ for $i \in I$.

**Step 1.** Randomly choose a facility $\hat{i} \in \bar{I}$. Set $y'_{\hat{i}} = 1$ remove $\hat{i}$ from $\bar{I}$. Proceed to Step 2.

**Step 2.** If $\bar{J} = \emptyset$, STOP with feasible solution. Else, randomly choose a customer $\hat{j} \in \bar{J}$.

- If $(a_{ij} + \ell_{ij}) \leq b_i$ set $x'_{ij} = 1, v'_{ij} = \ell_{ij}, b'_i = b'_i - (a_{ij} + \ell_{ij})$ and repeat Step 2.
- Otherwise, if $\bar{I} \neq \emptyset$, return to Step 1; else return to Step 0.

This procedure randomly assigns customers to a single randomly chosen facility as long as capacity allows. When sufficient capacity no longer exists, a new facility is opened and the procedure continues. This approach typically yields a solution with fewer open facilities than the first approach proposed. Randomly assigning customers to open facilities individually, on the other hand, appears to be less desirable than taking advantage of the GAPFD heuristic. However, in our computational testing, the solutions obtained via random generation contained a number of open facilities more consistent with the number found in the final solution generated by the neighborhood search heuristic. Therefore, the duration of the overall search procedure was reduced by choosing the random method.

Since the quality of the ultimate solution found by the search heuristic was unchanged by the method used to obtain the initial feasible solution, we use the random procedure in our computational testing.

### 6.3.2 FS Move Choice

In this section we determine the best set of moves to consider in our search and the order in which they should be considered. An obvious implementation considers opening (one at a time) all facilities in $C$ (called an $O$-move), closing (one at a time) all facilities in $O$ (called a $C$-move), and swapping all pairs of facilities in $O$ and $C$ during any single iteration of the search (called an $S$-move). We refer to this implementation as
the Full Neighborhood search (FNS). In this implementation, the order in which moves are considered is irrelevant. Intuitively, since the entire neighborhood is explored, a highly desirable heuristic solution is likely to result. However, the effort to search the full neighborhood requires extensive computational effort. In our computational study, we will analyze the advantages of solutions obtained from this implementation as well as the associated computing time.

Alternatively, we may consider only a subset of the full neighborhood at each iteration. This may lead to a reduction in the time needed to complete the search heuristic. However, the corresponding reduction in the size of the neighborhood explored has the potential for convergence to solutions of lesser quality. In the alternative implementation to follow we contend that the time saved through the search of a reduced neighborhood is not at the expense of solution quality. The direction for our second implementation results from of a careful study of the progress of the FNS implementation. A close analysis of the best moves at each iteration of FNS consistently revealed that the heuristic begins by choosing a sequence of O-moves. Then, at some iteration in the search procedure, the best move becomes an S-move. The S-move is continuously identified as the best move chosen until the search terminates without finding further improvement. Therefore, consideration of C-moves and S-moves during the first phase of the FNS-search equates to wasted time. Similarly, the consideration of O-moves and C-moves in the second phase of the FNS implementation is typically not beneficial.

Based on the preceding discussion, we divide our second implementation into two phases. Initially, we consider only O-moves. Since the number of O-moves to consider at any iteration is on the order of the number of facilities, |I|, we consider each of these alternatives. Recall that, to assess the benefit of each move, we solve a GAPFD to determine the corresponding customer assignments and the level at which their demand is fulfilled. The best incumbent is identified as the solution with the largest net profit. We repeat the open neighborhood search on this solution. Phase 1 continues until no
improvement is attained through an $O$-move. At this point, we proceed by searching the neighborhood of the best solution obtained in Phase 1, defined by potential swaps of open and closed facilities. Clearly, the number of potential $S$-moves to consider at each iteration is on the order of $|\mathcal{I}|^2$. Considering each of these moves is time prohibitive. Ahuja et al. [6] cite this concern as well, and identify two rules for assessing the expected impact of potential $S$-moves. Our rule is an extension of the rule proposed in that work, which reassigns all customers to the same new facility, while leaving the other customer assignments unchanged. That is, we consider each pair of currently open/closed sets $(o,c)$, $\forall o \in O$, $c \in C$. All customers currently assigned to facility $o$ are temporarily assigned to facility $c$. All customers assigned to facilities $i \in \mathcal{I}$; $i \neq o$ retain their current assignments.

For the updated collection of assignments, let $i_j$ denote the facility to which customer $j$ is assigned. Given this set of temporary assignments, we solve the following LP (SP$'$) to determine the optimal levels at which to fulfill those customers’ demands:

$$\theta + \max \sum_{j \in \mathcal{J}} r_{ij} w_{ij}$$

subject to

$$\sum_{j: i=j} w_{ij} \leq b_i - \sum_{j: i=j} (a_{ij} + \ell_{ij}) \quad i \in (O\{o\}) \cup \{c\}$$

$$0 \leq w_{ij} \leq u_{ij} - \ell_{ij} \quad j \in \mathcal{J},$$

where $\theta$ is a constant equal to $\sum_{j \in \mathcal{J}} (p_{ij} + r_{ij} \ell_{ij}) - \sum_{i \in (O\{o\}) \cup \{c\}} f_i$. It is easy to see that this problem decomposes into $|\{(O\{o\}) \cup \{c\}|$ continuous knapsack problems of the form SP, presented in Section 6.2.2. The optimal objective value of this problem, $z_{SP'}$, determines the ‘swap priority’ of a specific open/close pair, $(o,c)$. Specifically, after SP$'$ is solved for each $(o,c)$, $o \in O$, $c \in C$, the collection of pairs is sorted in non-increasing order of the corresponding value, $z_{SP'}$. 

121
At each iteration of our search we consider moves in the order determined by the preceding procedure until either (i) an improving move is found or (ii) no improving moves are found among the first \(|Z|\) swap candidates considered in the last iteration. In the case of (i), we proceed to the next iteration with the redefined neighborhood, while, in the case of (ii), the entire search procedure terminates. Comparison of this priority measure against the true desirability of moves determined by searching the full swap neighborhood revealed that our methodology was effective in identifying promising \(S\)-moves efficiently. We will address the effectiveness of our priority measure more directly in Section 6.4.

We have now completely described our facility neighborhood search implementation. However, we have not explicitly explained our omission of \(C\)-moves. Again, the basis for this implementation was that the full neighborhood very rarely found a \(C\)-move to be the best option at any iteration. We believe this result is due to the nature of the initial feasible solution provided. Since the randomly generated feasible solution aims to contain a small number of opened facilities, a \(C\)-move is intuitively unnecessary. On the contrary, when our initial solution was obtained by opening all facilities and solving the corresponding GAPFD, then only \(C\)-moves were chosen in the initial phase of FNS, and \(O\)-moves were rarely utilized. Therefore, whether \(O\)-moves and/or \(C\)-moves should be considered in Phase 1 depends upon the characteristics of the initial solution provided to the search heuristic by the decision maker.

Lastly, we are interested in the potential impact of the VLSN procedure, discussed in Section 6.2.2, when used in conjunction with the facility neighborhood search. Therefore, to improve upon the customer assignments associated with the set of procured facilities determined in the facility neighborhood search procedure, we will also test a hybrid implementation. This hybridization performs a single-customer multiple-exchange VLSN on the best solution found in the facility neighborhood search procedure. Of particular interest will be whether the improved solution merits the added computational expense.
With this consideration and the details of the proposed heuristic in place, we are ready to discuss a computational study designed to test the effectiveness of the variants of our heuristic when compared with the performance of a well-known commercial solver.

6.4 Computational Study

Our computational study considers twelve separate problem scenarios. For each problem scenario, we consider 10 separate instances. We vary the ratio of customers/facilities in order to explore the effect on both computational time requirements and solution quality. Our study assesses the performance of the three implementations discussed in Section 6.3.2. Additionally, we determine the time required for a state-of-the-art commercial solver to find solutions of equivalent quality to those obtained by our various implementations. The experiments were performed on a PC with a 3.40 GHz Pentium IV processor and 2 GB of RAM. All mixed-integer programming problems were solved using CPLEX 11.0.

6.4.1 Experimental Data

In our testing of the CFLFD, we consider instances with 15 and 30 facilities and a varying number of customers equal to $|J| = 3|I|$, 5|I|, 10|I|, 25|I|, 50|I|, and 100|I|. For each customer, we generate the random vectors of fixed profit parameters $P_j$ from uniform distributions on [30, 50] and the elements of the random vector of revenues, $R_j$, from a uniform distribution on [2, 5]. The vectors of customer requirements $A_j$, $L_j$ and $D_j$ are generated from uniform distributions on [10, 20], [75, 125], and [15, 35], respectively. Here $D_j$ is a random vector whose elements represent the range of acceptable sizes to fulfill customer demand, i.e., the upper bound on the demand for customer $j$, when assigned to facility $i$, is equal to $L_{ij} + D_{ij}$. Similar to the GAPFD, we generate identical facility capacities such that $b_i = \rho |\mathcal{J}| (i \in \mathcal{I})$, where

$$
\beta = \tau \cdot \frac{E(\min_{i \in \mathcal{I}} (A_{i1} + L_{i1}))}{|\mathcal{I}|},
$$

(6–11)
with $E$ referring to the expected value of the given expression. The parameter $\rho > 1$, inflates the capacity of a facility to ensure that not all facilities are required in a feasible solution to the CFLFD. In these experiments, the facility capacities were generated using $\rho = 2$. The parameter $\tau$ controls the level of flexibility available when determining the level at which each customer’s demand is fulfilled. These computational tests consider a moderate flexibility level by setting $\tau = 1.2$. We assume that the cost of procuring a facility is directly proportional to the capacity of the individual facility. Therefore, the cost of procuring facility $i$ ($i \in \mathcal{I}$) is given by $F_i = b_i C_i$, where $C_i$ represents the unit cost of procurement generated from a uniform distribution on $[0, 1.5]$.

As a side note, recall that at each step of the neighborhood search we effectively solve a GAPFD. In the heuristic proposed in Chapter 5 perturbation of resource capacities is necessary when customer-requirement parameters are facility dependent. For this reason, we perturb our set capacities using the procedure and parameter proposed in that chapter.

The tables in the following section assess the performance of the three implementations described in Section 6.3.2. Each row in the tables represents the average results collected amongst 10 instances generated for that particular scenario. The following measures are reported in Tables 6-1–6-4. A column labeled $FNS$ indicates that the full neighborhood (i.e., all swap, open, close moves) was considered at each iteration of the procedure. A column labeled $RNS$ indicates that the two-phase implementation was used. That is, we considered only $O$-moves in Phase 1 until no additional improvement was found, then $S$-moves in Phase 2, until the procedure terminated. Lastly, a column including a heading of hybrid indicates results obtained by running a single-customer multi-exchange VLSN on the solution obtained by either the FNS or RNS procedure. In addition, we use the following notation:

- **UB Error**: The upper bound on the error associated with the objective value of the solution obtained from the specified procedure. For example, in the case of FNS,

$$\text{UB error} = \frac{z^{UB} - z^{FNS}}{z^{FNS}}.$$
The upper bound on the optimal solution, $z^{UB}$, was obtained from CPLEX. The value was taken to be the best upper bound available after solving the CFLFD for 15 minutes.

- **Heuristic Procedure Time**: Computational time (in seconds) required by the specified procedure (i.e. FNS, RNS, or Hybrid).

- **CPLEX Time**: Computational time (in seconds) for CPLEX to obtain the same (or better) solution than that found by the specified procedure.

### 6.4.2 Results

As suggested in Section 6.2.1, the highest quality heuristic results are likely to result from the full neighborhood search implementation (FNS). Tables 6-1 and 6-3 provide average results for instances with 15 and 30 facilities, for the FNS, both with and without the supplementary VLSN step. These tables indicate that solutions with an average error no more than 4% were obtained for each of the problem sizes tested. The additional VLSN step (i.e., FNS hybrid) reduced the error only a small amount in each set of experiments. The most extreme improvements occur for experiments with a customer-to-facility ratio less than 10 for instances with 15 facilities, while the benefit of the VLSN step extends to instances with a customer-to-facility ratio up to 25 for instances with 30 facilities. The time required for FNS without VLSN ranged from approximately 2 to 90 seconds for 15 facility instances and 1 to 25 minutes for 30 facility instances. Table 6-1 shows that CPLEX required up to 10 times the amount of computational time to obtain the same solutions for 15 facility instances, while from Table 6-3 we see that CPLEX consistently outperformed the FNS implementation without the VLSN step for instances with 30 facilities. In fact, for instances with 30 facilities and 3000 customers, the FNS implementation, as well as CPLEX, required more than the allotted time of 30 minutes and therefore these results are not reported. The FNS hybrid time in Tables 6-1 and 6-3 indicates that the additional VLSN effort only marginally increased the total time for experiments with a ratio of customers to facilities no greater than 10. Unfortunately, as shown in the results for the 750 customer experiments, in Table 6-1, for larger instances,
the additional VLSN step becomes time prohibitive. In fact, the hybrid implementation for instances with 1500 customers or more takes more than the allotted run time to terminate; therefore, these results are omitted from Tables 6-1 and 6-3. Interestingly, while the improvement in solution quality in the hybrid implementation of FNS is minimal, the time required by CPLEX to obtain these slightly better solutions is consistently more than double the time required to obtain the solutions produced by FNS alone. Therefore, for instances with a small customer-to-facility ratio, the hybrid implementation is attractive. For these instances, the FNS hybrid run time remains small, but the quality of solutions is improved to a degree that the commercial solver has difficulty duplicating the result in a comparable amount of time.

The value of the two-phase reduced neighborhood search implementation (i.e., RNS) is presented in Tables 6-2 and 6-4, which offer average results collected over the same set of 15 and 30 facility instances considered in Tables 6-1 and 6-3. The average time for RNS, without VLSN, is at least 6 times less than FNS for 15 facility instances and at least 15 times less for 30 facility instances. More significantly, the average errors of the solutions obtained from RNS are only slightly higher than those obtained from FNS. In fact, for instances with 15 facilities and 150 customers or greater, the average error is within one one-hundredth of a percent of that obtained through FNS. The most extreme increase in error corresponds to 30 facility, 90 customer instances, where the error is approximately twice that obtained from the FNS implementation. However, for instances of this size, CPLEX obtains high quality solutions in a small amount of time; therefore, the need for the heuristic is less significant. The impact of VLSN applied to solutions obtained from RNS is similar to that seen when VLSN is combined with FNS. The most significant improvement in solution quality is seen with customer-to-facility ratios of 10 or less. As with the FNS hybrid, the RNS hybrid is exceedingly time consuming for both sets of instances with a customer-to-facility ratio greater than 10.
Because the RNS results provide comparably high quality solutions in less time, the ratio of time required for CPLEX and RNS to find the same solution is notably higher than the same ratio with respect to FNS. In addition, our study found that the average number of search iterations required by both the FNS and RNS implementations was remarkably similar in each set of experiments. This suggests that our 2-Phase RNS implementation does not terminate prematurely as a result of searching a limited neighborhood. Furthermore, recall that FNS considers all $S$-moves at each iteration, while no more than $|I|$ swaps are considered in the RNS implementation. Since the final solution obtained by each implementation is, on average, remarkably similar, we conclude that the most desirable swaps were considered in phase 2 of our RNS implementation. This is a strong indication that the swap ordering rule proposed in Section 6.3.2 is effective.

Lastly, Tables 6-1 and 6-2 suggest a small but noticeable increase in average heuristic error as the number of customers increases. It should be noted that errors were calculated using the best upper bound obtained by CPLEX after solving the CFLFD as a mixed-integer program for 15 minutes. It is expected that the difficulty of solving the CFLFD increases with an increase in the number of customers. Therefore, it is likely that the upper bound obtained after 15 minutes for instances with 1500 customers is weaker than the upper bound obtained for an instance with only 150 customers. This suggests that errors may be inflated as the number of customers increases, which is precisely what we observe in Tables 6-1 and 6-2.

### 6.5 CFLFD Heuristic Applications and Conclusions

The success of the heuristic framework proposed in this chapter is promising for optimization problems with a similar structure. Specifically, there are a number of familiar problems with a fixed-charge component that can be solved with our general framework. As mentioned previously, the CFLP clearly fits into our framework and an efficient method for solving the GAP subproblem is readily available. Our heuristic searches the
neighborhood of a CFLP solution in a different manner, with alternative move choices to those proposed by Ahuja et al. [6]. Specifically, the manner in which customer assignments are determined when evaluating potential moves relies predominantly on the reassignment of all customers through a secondary heuristic. This approach is pursued in lieu of searching the multi-customer exchange facility improvement graph and considering partial reassignments of customers in intermediary steps of the facility neighborhood search, as done for the CFLP in Ahuja et al. [6]. Our results suggest that this implementation choice provides quality solutions in a reasonable amount of time for problem sizes much larger than those considered in approaches for related problems. Therefore this methodology offers a unique alternative to solving this class of problems. In addition, the well-studied Uncapacitated Facility Location Problem (UFLP) (Erlenkotter [32], Sun [93]) fits well into our framework. For the UFLP, the subproblem to determine the assignments associated with a set of procured facilities can be trivially solved to optimality. In this case, our heuristic simplifies to a related approach proposed by Ghosh [45]. However, the order and subset of the neighborhood moves searched in our implementation is distinctly different. A final class of optimization problems which fits into our heuristic framework is the Fixed-Charge Transportation Problem. Various approaches have been proposed to solve this problem both heuristically (i.e., Adlakha and Kowalski [1]) and exactly (i.e., Gray [47]). Interestingly, if placed in our framework, the underlying subproblem solved to determine the value of each move is simply a linear transportation problem. The transportation problem itself has been well studied and solution methods have been presented in work originating with Ford and Fulkerson [36].

The lessons learned in this work with regard to implementation of our neighborhood search, can be applied directly to each of the additional optimization problems mentioned in this section. Our computational study illustrates than an intelligent search of a reduced facility neighborhood offers high quality solutions with a small amount of computational effort. Furthermore, for problems with a small ratio of customers to facilities, the
inclusion of a VLSN procedure improves the quality of the solutions obtained with minimal additional effort. However, our study also demonstrates that a single-customer multi-exchange heuristic is not practical for problems with a large number of customers. For these instances, our facility neighborhood search performs very well as a stand-alone heuristic. Therefore, our work offers an attractive heuristic which can be tailored to successfully solve a broad class of problem instances for both the CFLFD and similar fixed-charge problems.

Table 6-1. FNS results: 15 facilities

<table>
<thead>
<tr>
<th># Customers</th>
<th>FNS time (sec)</th>
<th>FNS error (%)</th>
<th>FNS time (sec)</th>
<th>CPLEX time (sec)</th>
<th>FNS hybrid time (sec)</th>
<th>FNS hybrid error (%)</th>
<th>CPLEX FNS hybrid time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>45</td>
<td>2.4</td>
<td>1.58</td>
<td>1.6</td>
<td>2.4</td>
<td>0.90</td>
<td>9.1</td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>3.9</td>
<td>0.81</td>
<td>45.8</td>
<td>4.1</td>
<td>0.39</td>
<td>722.7</td>
<td></td>
</tr>
<tr>
<td>150</td>
<td>9.2</td>
<td>0.66</td>
<td>34.8</td>
<td>10.9</td>
<td>0.52</td>
<td>292.7</td>
<td></td>
</tr>
<tr>
<td>375</td>
<td>33.9</td>
<td>0.40</td>
<td>90.0</td>
<td>78.9</td>
<td>0.37</td>
<td>267.3</td>
<td></td>
</tr>
<tr>
<td>750</td>
<td>35.0</td>
<td>1.42</td>
<td>160.9</td>
<td>576.2</td>
<td>1.41</td>
<td>301.6</td>
<td></td>
</tr>
<tr>
<td>1500</td>
<td>82.5</td>
<td>1.56</td>
<td>640.5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

Table 6-2. RNS results: 15 facilities

<table>
<thead>
<tr>
<th># Customers</th>
<th>RNS time (sec)</th>
<th>RNS error (%)</th>
<th>CPLEX RNS time (sec)</th>
<th>RNS hybrid time (sec)</th>
<th>RNS hybrid error (%)</th>
<th>CPLEX RNS hybrid time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>45</td>
<td>0.4</td>
<td>2.00</td>
<td>0.9</td>
<td>0.4</td>
<td>1.54</td>
<td>183.5</td>
</tr>
<tr>
<td>75</td>
<td>0.5</td>
<td>0.87</td>
<td>45.7</td>
<td>0.7</td>
<td>0.43</td>
<td>541.8</td>
</tr>
<tr>
<td>150</td>
<td>1.0</td>
<td>0.66</td>
<td>34.8</td>
<td>2.8</td>
<td>0.53</td>
<td>259.2</td>
</tr>
<tr>
<td>375</td>
<td>3.3</td>
<td>0.40</td>
<td>90.6</td>
<td>47.5</td>
<td>0.37</td>
<td>265.9</td>
</tr>
<tr>
<td>750</td>
<td>4.4</td>
<td>1.43</td>
<td>161.1</td>
<td>530.0</td>
<td>1.42</td>
<td>301.8</td>
</tr>
<tr>
<td>1500</td>
<td>9.9</td>
<td>1.56</td>
<td>642.0</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 6-3. FNS results: 30 facilities

<table>
<thead>
<tr>
<th># Customers</th>
<th>FNS time (sec)</th>
<th>FNS error (%)</th>
<th>FNS time (sec)</th>
<th>CPLEX FNS time (sec)</th>
<th>FNS hybrid time (sec)</th>
<th>FNS hybrid error (%)</th>
<th>CPLEX FNS hybrid time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td>74.7</td>
<td>3.86</td>
<td>3.0</td>
<td>75.2</td>
<td>2.73</td>
<td>25.5</td>
<td></td>
</tr>
<tr>
<td>150</td>
<td>153.1</td>
<td>2.20</td>
<td>4.3</td>
<td>155.9</td>
<td>1.22</td>
<td>230.6</td>
<td></td>
</tr>
<tr>
<td>300</td>
<td>219.4</td>
<td>1.33</td>
<td>10.0</td>
<td>281</td>
<td>0.63</td>
<td>1281.4</td>
<td></td>
</tr>
<tr>
<td>750</td>
<td>620.2</td>
<td>0.63</td>
<td>51.4</td>
<td>1794</td>
<td>0.28</td>
<td>1416.9</td>
<td></td>
</tr>
<tr>
<td>1500</td>
<td>1407.7</td>
<td>0.39</td>
<td>188.4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>3000</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

129
Table 6-4. RNS results: 30 facilities

<table>
<thead>
<tr>
<th># Customers</th>
<th>RNS time (sec)</th>
<th>RNS error (%)</th>
<th>CPLEX RNS time (sec)</th>
<th>RNS hybrid time (sec)</th>
<th>RNS hybrid error (%)</th>
<th>CPLEX RNS hybrid time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td>4.2</td>
<td>8.29</td>
<td>0.9</td>
<td>5.0</td>
<td>5.12</td>
<td>1.0</td>
</tr>
<tr>
<td>150</td>
<td>8.7</td>
<td>2.40</td>
<td>3.0</td>
<td>11.7</td>
<td>1.73</td>
<td>11.5</td>
</tr>
<tr>
<td>300</td>
<td>12.5</td>
<td>1.48</td>
<td>9.9</td>
<td>63.0</td>
<td>0.97</td>
<td>775.9</td>
</tr>
<tr>
<td>750</td>
<td>33.4</td>
<td>0.62</td>
<td>51.4</td>
<td>1261.5</td>
<td>0.28</td>
<td>1344.0</td>
</tr>
<tr>
<td>1500</td>
<td>76.3</td>
<td>0.39</td>
<td>185.5</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>3000</td>
<td>207.7</td>
<td>0.29</td>
<td>1660.3</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
CHAPTER 7
RESOURCE CONSTRAINED ASSIGNMENT PROBLEMS WITH SHARED RESOURCE CONSUMPTION

7.1 Introduction

In Chapters 4–6 we studied CFLFD and its special case, GAPFD. In both GAPFD and CFLFD the total amount of resource consumption, as well as revenue, is equal to the accumulation of each individual customer’s contribution to these two metrics. However, in almost any manufacturing scenario, certain product groups share similar production requirements. For example, fulfilling certain customers’ demands may require the same machine settings due to common components. Therefore, producing for these common customers on the same machine minimizes the total time required to satisfy their demands. The option of assigning customers to common facilities, at an added benefit, is considered in joint-cost assignment problems (Shubik [91]). Work in this area models the impact of joint rewards obtained from separate individuals who are willing to work as a group, as seen in the Mixed-Integer Setup Knapsack Problem proposed by Altay et al. [9]. Separately, problems that consider joint costs across a number of items are frequently found in multi-item inventory settings such as the Joint Replenishment Problem (Federgruen and Zheng [33]). However, the choice of how to utilize capacity saved by assigning customers with similar production requirements to the same machine is a separate feature of the optimization model and one that has received much less attention in these studies. More relevant work is found in scheduling problems that exploit the use of common resources when determining the release time of customers’ jobs. For example, Li [60] considered a resource constrained scheduling problem in which the amount of resource consumed is a function of the time in which the customers’ job is released. Conceivably, customers of similar types, released in uninterrupted sequence, require a lesser amount of resource. For problems with an assignment-based structure, Mazzola [68] considered a generalization of the GAP with nonlinear capacity interaction. Rather than accounting for some shared setup component of resource consumption, this work
modeled the interaction between customers assigned to the same facility. In contrast to the approach proposed in this chapter, Mazzola [68] provided a branch-and-bound algorithm that was shown to solve problems with up 20 customers.

Another interesting extension to more traditional assignment problems is the inclusion of varied forms of demand fulfillment constraints. In light of new restrictions in 21st century production, new forms of constraints need be considered in conjunction with traditional planning models. That is, rather than considering only limits on physical resources (or time), the planner must make decisions that adhere to additional limits placed on the satisfaction of customer demand. For example, as mentioned previously, customers may be grouped into types, with each type interpreted as either (i) a specific client, (ii) common production requirements, (iii) a common shipment destination or (iv) common manufacturing byproducts. Given these possible interpretations, the level at which customer demand is fulfilled may be limited by either transportation resources or physical space allocated in the warehouse facility. Alternatively, with growing concern on manufacturers’ impacts on the environment, one must consider how production associated with fulfilling customer demand may result in either pollution or hazardous materials. That is, the total amount of demand satisfied within a particular group (or type) may be limited by outside parties, such as the Environmental Protection Agency or the Occupational Safety and Health Administration. These are specific examples that are common across a number industries. Thus, it is clear that it is important to include constraints that limit production across all resources for a given subset of customers. The models studied in the preceding chapters do not account for this additional form of production limitations. It is important to note that few other works in the literature consider this problem element either. Loosely related problems that have received more attention in the literature are those which consider multiple resource consumption. A specific example is found in the Multi-resource Generalized Assignment Problem, considered by Gavish and Pirkul [40], among others. This model assumes that each facility
consumes multiple resources to satisfy a customer’s demand. However, limits on capacity consumption do span customers assigned to different agents. Separately, Mazzola and Neebe [69] study the Resource-Constrained Assignment Scheduling Problem (RCAS). This extension of the pure assignment problem considers a side constraint that restricts the total consumption of a resource amongst decentralized customer and facility sets. It is the limitation of demand fulfillment for customers assigned to numerous facilities that we wish to account for in the model considered in this chapter.

As briefly alluded to in the discussion of relevant literature, this chapter considers a new class of problems that, among other things, model production environments in which a portion of capacity consumption is shared among common customers satisfied by that resource. As in the GAPFD and CFLFD, the new optimization model assigns customers to facilities and determines the corresponding demand fulfillment levels. However, different considerations must be taken into account when making these decisions. First, customers are now grouped by type. Customers of the same type, assigned to the same facility, consume a fixed amount of resource in addition to their individual consumption. Also, in addition to the resource limitations of individual facilities, aggregate resource consumption among customers of a particular type is subject to a separate set of restrictions. This model adds a level of complexity to the decision-making process. A planner must now consider the impact of saving capacity by assigning customers of the same type to the same facility. Furthermore, the additional limitation of production associated with a particular customer set may affect the levels at which demands are satisfied. Since this problem still considers the assignment of customers to facilities, we again pursue a branch-and-price approach based on a reformulation of our model. However, unlike Chapter 4, the so-called master problem is no longer in the form of the set-partitioning problem. Due to the additional capacity constraints, the master problem, pricing problem, and column representations must be carefully developed. We show that the pricing
problem belongs to an unstudied class of knapsack problems for which we propose efficient solution approaches.

The remainder of this chapter is organized as follows. Section 7.2 presents the optimization model which we consider. Sections 7.3 and 7.4 develop an exact algorithm for a special case of the model with fixed customer demand. Then, Section 7.5 considers the variant of the model with flexible demand and derives an important equivalent formulation of the problem for which an exact algorithm is derived. Section 7.7 provides details of the implementation of our branch-and-price algorithm. Finally, Section 7.8 discusses a computational study of our approach.

7.2 Model Formulation

We consider a set of customers \( J \). Each customer’s demand \( j (j \in J) \) must be satisfied by a single facility \( i (i \in I) \). If customer \( j (j \in J) \) is assigned to facility \( i (i \in I) \), then a fixed profit, \( p_{ij} \), is accrued and a fixed amount of capacity \( a_{ij} \) is consumed. Furthermore, the corresponding customer demand fulfillment level must be selected from the interval \([\ell_{ij}, u_{ij}]\) and an additional profit \( r_{ij} \) is accrued per unit of demand fulfillment. Each facility \( i \in I \) has capacity \( b_i \ (i \in I) \). In addition to these considerations, we have a set of customer types \( Q \), where each customer is associated with a single type. Customers of type \( q \ (q \in Q) \) belong to the set \( J_q \). The first distinguishing characteristic of this model is defined by the manner in which resource consumption may be shared. If any customer of type \( q \ (q \in Q) \) is assigned to facility \( i \), a fixed amount of resource, \( f_{iq} \), is consumed. Furthermore, the collective capacity consumed by customers of type \( q \) is limited by \( g_q \ (q \in Q) \). The objective is to determine the assignment of customers to facilities, as well as the corresponding customers demand fulfillment levels, in order to maximize total profit, while satisfying the capacity constraints of the facilities and the individual customer types. The model we consider is then given by

\[
\text{maximize} \sum_{i \in I} \sum_{j \in J} (p_{ij} x_{ij} + r_{ij} v_{ij})
\]
subject to (FASR)

\[
\sum_{q \in Q} f_{iq} \max_{j \in J_q} x_{ij} + \sum_{j \in J} v_{ij} \leq b_i \quad i \in \mathcal{I} \quad (7-1)
\]

\[
\sum_{i \in \mathcal{I}} \sum_{j \in J_q} v_{ij} \leq g_q \quad q \in \mathcal{Q} \quad (7-2)
\]

\[
\sum_{i \in \mathcal{I}} x_{ij} = 1 \quad j \in \mathcal{J} \quad (7-3)
\]

\[
\ell_{ij} x_{ij} \leq v_{ij} \leq u_{ij} x_{ij} \quad i \in \mathcal{I}; \ j \in \mathcal{J} \quad (7-4)
\]

\[
x_{ij} \in \{0, 1\} \quad i \in \mathcal{I}, \ j \in \mathcal{J} \quad (7-5)
\]

which we refer to as the **Flexible Demand Resource Allocation Problem with Shared Resource Considerations**. Constraints (7–1) require that the capacity (variable and shared) consumed by customers assigned to facility \(i\) is no greater than \(b_i\) \((i \in \mathcal{I})\). Furthermore, constraints (7–2) ensure that demand satisfied for customers of type \(q\) does not exceed \(g_q\) \((q \in \mathcal{Q})\). Constraints (7–3) and (7–4) are the assignment and flexibility constraints introduced in Chapter 3. For sake of simplicity, and without loss of generality, we’ve assumed that the individual customer fixed capacity consumptions \(a_{ij}\) \((i \in \mathcal{I}; \ j \in \mathcal{J})\) are included in the demand level bounds. That is, both \(\ell_{ij}\) and \(u_{ij}\) are increased by the amount \(a_{ij}\) \((i \in \mathcal{I}; \ j \in \mathcal{J})\). This implies that the fixed profit parameters \(a_{ij}\) are reduced by the amount \(r_{ij} a_{ij}\) \((i \in \mathcal{I}; \ j \in \mathcal{J})\).

Notice that FASR is a mixed integer program with a set of non-linear constraints and linear objective function. The non-linear constraints, (7–1) can be linearized as shown in the following formulation.

\[
\text{maximize} \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} (p_{ij} x_{ij} + r_{ij} v_{ij})
\]

subject to (FASR’)

\[
\sum_{q \in \mathcal{Q}} f_{iq} s_{iq} + \sum_{j \in \mathcal{J}} v_{ij} \leq b_i \quad i \in \mathcal{I}
\]
\[
\sum_{i \in I} \sum_{j \in J} v_{ij} \leq g_q \quad q \in Q
\]

\[
\sum_{i \in I} x_{ij} = 1 \quad j \in J
\]

\[
\ell_{ij} x_{ij} \leq v_{ij} \leq u_{ij} x_{ij} \quad i \in I; j \in J
\]

\[
s_{iq} \geq x_{ij} \quad i \in I; j \in J_q; q \in Q \quad (7-6)
\]

\[
x_{ij} \in \{0, 1\} \quad i \in I; j \in J
\]

The constraints (7–6) ensure that if a customer of type \( q \) \( (q \in Q) \) is assigned to facility \( i \) \( (i \in I) \), then the a fixed amount of resource, \( f_{iq} \), is consumed. It should be noted that the alternative formulation does not prevent some percentage of the \( f_{iq} \) units of resource \( i \) from being consumed, even when no customers of type \( q \) are assigned to facility \( i \) \( (i \in I; q \in Q) \). In other words, no constraint prevents \( s_{iq} \) \( (i \in I; q \in Q) \) from being greater than zero even when \( x_{ij} = 0 \) for all \( j \in J_q \). It is interesting to note that this scenario arises only if the optimal solution is such that (i) all customers assigned to facility \( i \) have their demand satisfied at their upper bounds, or (ii) no customers are assigned to facility \( i \) \( (i \in I) \) in an optimal solution. However, in each of these cases, the objective value and corresponding decision variables are still optimal to the original FASR model. Therefore FASR’ is an equivalent representation of FASR.

The purpose of this chapter is to develop a branch-and-price algorithm for FASR. As stated in previous chapters, the success of a branch-and-price approach relies strongly on the ability to effectively solve the corresponding pricing problem. To this end, we focus much of our attention on developing efficient solution methods for the resulting pricing problem. The development of these approaches is most easily presented by first considering a special case of FASR. Therefore, in Section 7.3 we initially develop an exact solution procedure for the non-flexible variant of FASR in which \( \ell_{ij} = u_{ij} = a_{ij} \) \( (i \in I; j \in J) \). We refer to this special case of the FASR as the Resource Constrained Assignment Problem with Shared Resource Consumption (RCAS).
maximize \( \sum_{i \in I} \sum_{j \in J} (\bar{p}_{ij} x_{ij}) \)

subject to (RCAS)

\[ \sum_{q \in Q} f_{iq} s_{iq} + \sum_{j \in J} a_{ij} x_{ij} \leq b_i \quad i \in I \quad (7-7) \]

\[ \sum_{i \in I} \sum_{j \in J} a_{ij} x_{ij} \leq g_q \quad q \in Q \]

\[ \sum_{i \in I} x_{ij} = 1 \quad j \in J \]

\[ s_{iq} \geq x_{ij} \quad i \in I; \quad j \in J_q; \quad q \in Q \]

\[ x_{ij} \in \{0, 1\} \quad i \in I; \quad j \in J \]

where \( \bar{p}_j = p_j + r_j a_j \) (\( j \in J \)).

7.3 Exact Algorithm for RCAS

In this section we propose a branch-and-price algorithm for RCAS based on an interesting reformulation of the problem. First we will need the following definitions and terminology.

- \( x^d_i \) is a binary vector with elements \( x^d_{ij} \) (\( j \in J \)) representing the \( d \)th subset of customers that can be feasibly assigned to facility \( i \) (with respect to constraint (7–7));
- \( D_i \) is the total number of subsets of customers that can be assigned to facility \( i \);
- \( \lambda^d_i \) is a binary variable with value 1 if the \( d \)th subset associated with facility \( i \) is used, and 0 otherwise;
- \( \alpha_i(x^d_i) = \sum_{j \in J} \bar{p}_{ij} x^d_{ij} \) (\( d = 1 \ldots D_i \); \( i \in I \)).
- \( \beta^d_i(x^d_i) = \sum_{q \in Q} a^d_{ij} x^d_{ij} \) (\( d = 1 \ldots D_i \); \( i \in I; \quad q \in Q \)).

The master problem in this case can be written as

maximize \( \sum_{i \in I} \sum_{d=1}^{D_i} \alpha_i(x^d_i) \lambda^d_i \)
subject to (MP)

\[
\sum_{i \in I} \sum_{d=1}^{D_i} \beta_i^q (x_i^d) \lambda_i^d \leq g_q \quad q \in Q
\]  

(7–8)

\[
\sum_{i \in I} \sum_{d=1}^{D_i} x_{ij}^d \lambda_i^d = 1 \quad j \in J
\]  

(7–9)

\[
\sum_{d=1}^{D_i} \lambda_i^d = 1 \quad i \in I
\]  

(7–10)

\[
\lambda_i^d \in \{0, 1\} \quad d = 1, \ldots, D_i; \ i \in I
\]

Next we define the following notation:

- \( \mu_q (q \in Q) \) are the dual variables associated with the customer set capacity constraint, (7–8) in (MP);
- \( \pi_j (j \in J) \) are the dual variables associated with constraints, (7–9) in (MP);
- \( \delta_i (i \in I) \) are the dual variables associated with the convexity constraints, (7–10), in (MP).

The pricing problem associated with resource \( i \) \((i = 1, \ldots, I)\) is now written as

\[
\text{maximize} \sum_{j \in J} (\bar{p}_{ij} - \pi_j - \mu_q a_{ij}) x_j - \delta_i
\]

subject to (PP\(_i\))

\[
\sum_{q \in Q} f_q \max_{j \in T_q} \{x_j\} + \sum_{j \in J} a_{ij} x_j \leq b_i
\]

\[
x_j \in \{0, 1\} \quad j \in J
\]

where \( q_j \) is the customer set to which customer \( j \) \((j \in J)\) belongs. An effective method for solving the pricing problem is instrumental in developing an effective branch-and-price procedure. Therefore, Section 7.4 studies a class of optimization problems that includes PP\(_i\).
7.4 Shared Consumption Knapsack Problem

In this section, we develop both heuristic and exact approaches for solving the following class of knapsack problems.

\[
\text{maximize } \sum_{j \in J} p_j x_j
\]

subject to

\[
\sum_{q \in Q} f_q s_q + \sum_{j \in J} a_j x_j \leq b \tag{7–11}
\]

\[
s_q \geq x_j \quad j \in J_q; \; q \in Q \tag{7–12}
\]

\[
x_j \in \{0, 1\} \quad j \in J. \tag{7–13}
\]

\[
s_q \in \{0, 1\} \quad q \in Q. \tag{7–14}
\]

In SKP, the binary restrictions (7–14) are clearly redundant. Therefore, they are not included in the following relaxation of SKP, which we refer to as SKPR,

\[
\text{maximize } \sum_{j \in J} p_j x_j
\]

subject to

\[
\sum_{q \in Q} f_q s_q + \sum_{j \in J} a_j x_j \leq b \tag{SKPR}
\]

\[
s_q \geq x_j \quad j \in J_q; \; q \in Q \tag{7–15}
\]

\[
x_j \in [0, 1] \quad j \in J. \tag{7–15}
\]

In the above relaxation, we relax the binary restrictions on the assignment variables \(x_j (j \in J)\) by replacing (7–13) with (7–15). Before continuing, it is worthwhile to note that there clearly exists an optimal solution to both SKP and SKPR for which \(x_j^{\text{SKP}} = 0\) and \(x_j^{\text{SKPR}} = 0\) if \(p_j \leq 0\). Therefore, the following assumption holds without loss of generality.
Assumption 2. For all $j \in \mathcal{J}$, $p_j > 0$.

To begin to motivate an approach to solve SKP, consider an equivalent non-linear formulation of SKP, SKP$'$.

$$\text{maximize } \sum_{j \in \mathcal{J}} p_j y_j$$

subject to

$$\sum_{q \in \mathcal{Q}} \left\{ f_q + \sum_{j \in \mathcal{J}_q} a_j y_j \right\} s_q \leq b$$  \hspace{1cm} (SKP$'$)

$$s_q \geq y_j \hspace{1cm} j \in \mathcal{J}_q; \ q \in \mathcal{Q}$$

$$y_j \in \{0, 1\} \hspace{1cm} j \in \mathcal{J}$$

$$s_q \in \{0, 1\} \hspace{1cm} q \in \mathcal{Q}$$

The alternative decision variables, $y_j \ (j \in \mathcal{J})$ are introduced to distinguish between the assignment variables in the alternative presentations of SKP. Clearly, in the case of SKP and SKP$'$, there exists an optimal solution in which $x_j = y_j \ (j \in \mathcal{J})$. However, this relationship does not necessarily hold when comparing the optimal assignment values $x_j \ (j \in \mathcal{J})$ obtained in SKPR to the values $y_j \ (j \in \mathcal{J})$ found in the following relaxation of SKP$'$, which we refer to as SKPR$'$.

$$\text{maximize } \sum_{j \in \mathcal{J}} p_j y_j$$

subject to

$$\sum_{q \in \mathcal{Q}} \left\{ f_q + \sum_{j \in \mathcal{J}_q} a_j y_j \right\} s_q \leq b$$  \hspace{1cm} (SKPR$'$)

$$s_q \geq y_j \hspace{1cm} j \in \mathcal{J}_q; \ q \in \mathcal{Q}$$

$$y_j \in [0, 1] \hspace{1cm} j \in \mathcal{J}$$

$$s_q \in \{0, 1\} \hspace{1cm} q \in \mathcal{Q}.$$
Note that in SKPR′ the binary restrictions on the assignment variables \( y_j \) (\( j \in J \)) are relaxed. However, the binary restrictions on the variables \( s_q \) (\( q \in Q \)) remain.

Interestingly, SKPR′ can be reformulated as a KPEI studied in Chapter 4. In this equivalent formulation, flexibility variables \( w_q \) (\( q \in Q \)) represent the collective amount of capacity consumed by all customers of type \( q \) (\( q \in Q \)). This equivalent formulation can be written as

\[
\text{maximize } \sum_{q \in Q} \tilde{r}_q(w_q)
\]

subject to

\[
\sum_{q \in Q} w_q \leq b \tag{7–17}
\]

\[
w_q \leq u'_q s_q \quad q \in Q \tag{7–18}
\]

\[
w_q \geq \ell'_q s_q \quad q \in Q \tag{7–19}
\]

\[
s_q \in \{0, 1\} \quad q \in Q \tag{7–20}
\]

where \( \ell'_q = f_q \) and \( u'_q = \left( f_q + \sum_{j \in J_q} a_j \right) \) (\( q \in Q \)), and \( \tilde{r}_q(w_q) \) is an optimal solution to the following parametric optimization problem, CKP,

\[
\text{maximize } \sum_{j \in J} p_j y_j
\]

subject to

\[
f_q + \sum_{j \in J_q} a_j y_j \leq w_q
\]

\[
y_j \in [0, 1] \quad j \in J_q,
\]

if \( w_q > 0 \), and 0 otherwise, for any feasible \( w_q \) (\( q \in Q \)) to KPEI. Note that, without loss of generality, we can also let \( \ell'_q = \min\{f_q, b\} \) and \( u'_q = \min\left\{ \left( f_q + \sum_{j \in J_q} a_j \right), b \right\} \) (\( q \in Q \)). For convenience, we retain this latter set of definitions throughout the remainder.
of the chapter. Now, notice that CKP is the continuous knapsack problem. Therefore, the function \( r_q(w_q) \) is readily obtained. First, let \( S_q \) be the set of customers \( j \in J_q \) with \( y^\text{CKP}_j > 0 \) in an optimal solution to CKP with \( w_q = u'_q \). (It should be noted that there is at most one \( j \in J_q \) for with \( 0 < y^\text{CKP}_j < 1 \). This will impact later structural results.) Now let \( \tilde{S}_q \) be an ordered set comprised of only customers in \( S_q \), in non-increasing order of \( p_ja_j \). Let \( k_q \) index the customers in \( \tilde{S}_q \) such that \( \arg \max_{j \in \tilde{S}_q} p_ja_j \) is associated with the index \( k_q = 0 \) and \( \arg \min_{j \in \tilde{S}_q} \frac{p_j}{a_j} \) with \( k_q = \lvert \tilde{S}_q \rvert - 1 \). Furthermore, let \( j_{k_q} \) be the customer in position \( k_q \) of \( \tilde{S}_q \). Then \( \tilde{p}_{k_q} = \sum_{k_{q}=0}^{k_q} p_{j_{k_{q}}} \) is the collective profit of the first \( k_q + 1 \) customers in \( \tilde{S}_q \) \((q \in Q)\) and \( \tilde{a}_{k_q} = \min \left\{ \left( f_q + \sum_{k_{q}=0}^{k_q} a_{j_{k_{q}}} \right), b \right\} \) is the total capacity consumed by the first \( k_q + 1 \) customers in \( \tilde{S}_q \). Using this notation, the function \( \tilde{r}_q(w_q) \) \((q \in Q)\) is characterized in the following lemma.

**Lemma 6.** In the optimization problem KPEI, the objective function \( \tilde{r}_q(w_q) \) \((q \in Q)\) is given by

\[
\tilde{r}_q(w_q) = \begin{cases} 
0 & w_q = 0 \\
-\infty & 0 < w_q < \tilde{a}_0 \\
\tilde{p}_{k_q-1} + \frac{p_{j_{k_q}}(w_q-\tilde{a}_{k_q-1})}{\tilde{a}_{k_q}} & \tilde{a}_{k_q-1} \leq w_q \leq \tilde{a}_{k_q}, \quad k_q = 0, \ldots, \lvert \tilde{S}_q \rvert - 1
\end{cases}
\]

where \( \tilde{a}_{(-1)} = f_q \) and \( \tilde{p}_{(-1)} = 0 \).

**Proof.** The function \( \tilde{r}_q(w_q) \) \((q \in Q)\), as defined in the lemma, is comprised of \( \lvert \tilde{S}_q \rvert \) linear segments \((q \in Q)\). The slope and interval of each linear segment is given by the per unit profit, \( \frac{p_j}{a_j} \), and the resource consumption level, \( a_j \), respectively, for a particular customer \( j \). The function is constructed such that segments are included in non-increasing order of the ratio \( \frac{p_j}{a_j} \) \((j \in S_q)\). From Kolesar [54], we know that an optimal solution to the continuous knapsack problem exists for which customers are included in order of their \( \frac{p_j}{a_j} \) values. Therefore, by construction, \( \tilde{r}_q(w_q) \) as defined in the lemma yields the optimal objective value to CKP. □
Both the exact and heuristic methodologies to solve SKP are motivated by a relaxation of KPEI studied in Chapter 4 denoted by RKP. For the specially structured revenue functions in this chapter, we refer to this relaxation as KPEIR.

\[
\begin{align*}
\text{maximize} & \quad \sum_{q \in Q} \theta_q(w_q) \\
\text{subject to} & \quad \sum_{q \in Q} w_q \leq b \\
& \quad w_q \geq 0 \quad q \in Q.
\end{align*}
\]

(KPEIR)

The function \(\theta_q (q \in Q)\) is the non-decreasing concave envelope encompassing the origin, the function \(\tilde{r}_q\) and the point \((b, \tilde{r}_q(u'_q))\). The profit functions, \(\theta_q(w_q) (q \in Q)\), are defined in the following lemma. First, let

\[
\begin{align*}
\hat{k}_q &= \inf \left\{ k_q = 0, \ldots, |\tilde{S}_q| - 1 : \frac{\sum_{k'=0}^{k_q} p_{jk_k'}}{f_q + \sum_{k'=0}^{k_q} a_{jk_k'}} \geq \frac{p_{jk_q+1}}{a_{jk_q+1}} \right\} \\
\end{align*}
\]

be the smallest index for which the collective per unit profit for customers associated with \(k_q \leq \hat{k}_q\) is no less than the per unit profit associated with customer \(\hat{j}_{\hat{k}_q+1}\) (with \(\frac{p_{j|\tilde{S}_q|}}{a_{j|\tilde{S}_q|}} = 0\)).

**Lemma 7.** In KPEIR, the functions \(\theta_q (q \in Q)\) are given by

\[
\theta_q(w_q) = \begin{cases} \\
\frac{\tilde{p}_{k_q}}{\tilde{a}_{k_q}} w_q & 0 \leq w_q \leq \tilde{a}_{k_q} \\
\tilde{r}_q(w_q) & \tilde{a}_{k_q} < w_q \leq u'_q \\
\tilde{r}_q(u'_q) & u'_q < w_q \leq b
\end{cases}
\]

Proof. For \(\theta_q\) to be a non-decreasing concave envelope encompassing the point \((0,0)\), the function \(\tilde{r}_q\) and the point \((b, \tilde{r}_q(u'_q))\), the following general description of \(\theta_q\) must hold.
\[
θ_q(w_q) = \begin{cases} 
\left( \frac{\tilde{r}_q(\mu_q)}{\mu_q} \right) w_j & 0 \leq w_j \leq \mu_j \\
\tilde{r}_q(w_q) & \mu_q < w_j \leq u'_j \\
\tilde{r}_q(u'_q) & u'_j < w_j \leq b,
\end{cases}
\] (7–23)

where

\[μ_q = \inf \left\{ t : t'_q \leq t \leq u'_q \text{ and } \frac{\tilde{r}_q(t)}{t} \in \partial \tilde{r}_q(t) \right\} \quad q \in Q.\]

Since \(\tilde{r}_q(w_q)\) is piecewise linear, \(μ_q\) clearly occurs at some breakpoint in \(\tilde{r}_q(w_q)\). Recall from Lemma 6 that each linear segment of \(\tilde{r}_q(w_q)\) corresponds to a customer in \(S_q\). Therefore, for some breakpoint \(k_q\) (\(k_q = 0 = 1, \ldots, |S_q| - 1\)) the slope of the segment connecting the origin to point \((\tilde{a}_{k_q}, \tilde{r}_q(\tilde{a}_{k_q}))\) is \(\frac{\tilde{p}_{k_q}}{\tilde{a}_{k_q}}\). By the definition of \(\tilde{r}_q(w_q)\) and the fact that \(θ_q(w_q)\) must be a non-decreasing concave envelope, the value \(μ_q\) is the smallest \(k_q\) (\(k_q = 0 = 1, \ldots, |S_q| - 1\)) for which

\[\frac{\tilde{p}_{k_q}}{\tilde{a}_{k_q}} \geq \frac{p_{j_{k_q + 1}}}{a_{j_{k_q + 1}}}.\]

Thus, the \(θ_q(w_q)\) in the lemma follows by the definition of \(k_q\) in (7–22).

Figure 7-1 illustrates an example of the two functions, \(r_q\) and \(θ_q\) (\(q \in Q\)) defined in Lemmas 6 and 7. In this example, \(σ_1 = \frac{p_{j_{k_1}}}{a_{j_{k_1}}}, σ_2 = \frac{p_{j_{k_2}}}{a_{j_{k_2}}}, σ_3 = \frac{p_{j_{k_3}}}{a_{j_{k_3}}}\) and \(σ_D = \frac{\sum_{k=1}^{k_2-1} p_{j_{k}}}{\sum_{k=1}^{k_2} a_{j_{k}}}\) are the slopes of the indicated segments. In this example, \(k_q = 2\).

Bretthauer and Shetty [19] propose an algorithm for a class of knapsack problems with general revenue functions that includes KPEIR. This approach consists of a binary search to obtain the optimal Lagrange multiplier associated with (7–21), which satisfies the KKT conditions of KPEIR. However, since \(θ_q(w_q)\) (\(q \in Q\)) is known to be piecewise linear, we can exploit this structure to solve the problem in an alternative manner. Let \(J^o\) be the set of ‘dummy customers’ representing the linear segments connecting the origin to
\((\hat{a}_{k_q}, \tilde{r}_q(\hat{a}_{k_q}))\) \((q \in \mathbb{Q})\). In addition, let \(J'_q = \{j_{k_q} \in \tilde{S}_q : k_q = k_q, \ldots, |\tilde{S}_q| - 1\} \) \((q \in \mathbb{Q})\). Then KPEIR is equivalently represented by KPEIR'.

\[
\text{maximize} \quad \sum_{j \in J^o} \left( \frac{\hat{p}_j}{\hat{a}_j} \right) w_j + \sum_{q \in \mathbb{Q}} \sum_{j \in J'_q} \left( \frac{\hat{p}_j}{\hat{a}_j} \right) w_j \\
\text{subject to} \quad (\text{KPEIR}') \\
\begin{align*}
w_j & \in [0, \hat{a}_j] & j & \in J^o & (7-24) \\
w_j & \in [0, \hat{a}_j] & j & \in J' \\
w_j & \geq 0 & j & \in J^o \\
w_j & \geq 0 & j & \in J^o
\end{align*}
\]

where

\[
\hat{p}_j = \begin{cases} 
\tilde{p}_{k_{\hat{q}_j}} & \text{if } j \in J^o \\
\tilde{p}_j & \text{if } j \in J' 
\end{cases} \\
(7-25)
\]

and

\[
\hat{a}_j = \begin{cases} 
\tilde{a}_{k_{\hat{q}_j}} & \text{if } j \in J^o \\
\tilde{a}_j & \text{if } j \in J'. 
\end{cases} \\
(7-26)
\]

and \(\hat{q}_j\) is the customer type that a particular ‘dummy customer’ \(j \in J^o\) represents. Furthermore, the parameter \(\hat{a}_j\) represents the amount of capacity consumed and profit accrued by customer \(j\) in the optimal solution to the parametric optimization problem CKP with \(w_q = u'_q\). Again, there is at most one customer within each type for which

\[
\hat{a}_j \neq a_j \text{ and } \tilde{p}_j \neq p_j. \\
(7-27)
\]
Then, given an optimal solution to $\text{KPEIR'}$, the optimal solution to $\text{KPEIR}$ is given by

$$w_{q}^{\text{KPEIR}} = \left( \sum_{j \in J'_q} w_{j}^{\text{KPEIR'}} + w_{\tilde{j}_q}^{\text{KPEIR'}} \right) q \in Q$$

where $\tilde{j}_q = \{ j \in J^o : q_j = q \}$.

Clearly, $\text{KPEIR'}$ is solvable by considering customers $j \in \{ J^o \cup \{ \cup_{q \in Q} J'_q \} \}$ in non-increasing order of $\frac{\hat{p}_j}{\hat{a}_j}$. Therefore, we are prepared to formally provide an algorithm for $\text{KPEIR}$.

**KPEIR Algorithm**

**Step 0.** Set $w_{q}^{\text{KPEIR}} = 0$ for $q \in Q$. Find $\tilde{k}_q$ and establish sets $J^o$, $J'_q (q \in Q)$ and

$$\tilde{J} = \{ J^o \cup \{ \cup_{q \in Q} J'_q \} \}.$$  

**Step 1.** Sort customers $j \in \tilde{J}$ in non-increasing order of $\frac{\hat{p}_j}{\hat{a}_j}$.

**Step 2.** Let $j$ be the first customer in $\tilde{J}$ and $\hat{q} = \{ q \in Q : j \in J'_q \text{ or } \tilde{q}_j = q \}$ the set associated with $j$. Set

$$w_{\hat{q}}^{\text{KPEIR}} = w_{q}^{\text{KPEIR}} + \min \{ b, \hat{a}_j \}$$

$$b = b - \min \{ b, \hat{a}_j \}.$$  

Set $\tilde{J} = \tilde{J}\setminus\{j\}$. If $b = 0$ or $\tilde{J} = \emptyset$, STOP, else repeat Step 2.

Using this algorithm we can efficiently solve $\text{KPEIR}$. The important structural property of $\text{KPEIR}$ from Lemma 2 in Chapter 4 motivates both a heuristic and customized branch-and-bound to solve $\text{SKP}$. This result is restated in the following lemma.

**Lemma 8.** An optimal solution to $\text{KPEIR}$ exists with at most one customer type $q$ such that

(i) $w_q < \ell'_q$, or

(ii) $\theta_q(w_q) > \tilde{r}_q(w_q)$ and $\ell'_q \leq w_q \leq \ell'_q$.

**Proof.** See proof of Lemma 2.
To understand how KPEIR and its structure allow us to effectively solve SKP, consider the relationship between SKPR and KPEIR given in the following theorem.

**Theorem 9.** SKPR and KPEIR are equivalent.

**Proof.** First, let \( w_{q}^{KPEIR} \) be an optimal solution to KPEIR. Consider constructing a feasible solution to SKPR in the following manner, for all \( q (q \in \mathcal{Q}) \).

If \( w_{q}^{KPEIR} = 0 \),

\[
\begin{align*}
    s_{q}^{SKPR} &= 0 \\
    x_{j}^{SKPR} &= 0 \quad \forall j \in \mathcal{J}_{q}.
\end{align*}
\]

Else if \( 0 < w_{q}^{KPEIR} < \tilde{a}_{k_{q}} \), let \( \tilde{x} = \frac{w_{q}^{KPEIR}}{\tilde{a}_{k_{q}}} \), then

\[
\begin{align*}
    s_{q}^{SKPR} &= \tilde{x} \\
    x_{j k_{q}}^{SKPR} &= \tilde{x} \left( \frac{\tilde{a}_{j k_{q}}}{a_{j k_{q}}} \right) \quad k_{q} = 0, \ldots, \hat{k}_{q} \\
    x_{j k_{q}}^{SKPR} &= 0 \quad \forall j : k_{q} = \hat{k}_{q} + 1, \ldots, |\tilde{S}_{q}| - 1 \\
    x_{j}^{SKPR} &= 0 \quad \forall j \in \{ \mathcal{J}_{q}/\tilde{S}_{q} \}.
\end{align*}
\]

Lastly, if \( \tilde{a}_{k} \leq w_{q}^{KPEIR} < \tilde{a}_{k+1} \) for some \( \hat{k}_{q} \geq k_{q} \),

\[
\begin{align*}
    s_{q}^{SKPR} &= 1 \\
    x_{j k_{q}}^{SKPR} &= 1 \quad k_{q} = 0, \ldots, \hat{k}_{q} \\
    x_{j (k_{q}+1)}^{SKPR} &= \frac{w_{q}^{KPEIR} - \tilde{a}_{k_{q}}}{a_{j (k_{q}+1)}} \\
    x_{j k_{q}}^{SKPR} &= 0 \quad k_{q} = \hat{k}_{q} + 2, \ldots, |\tilde{S}_{q}| - 1 \\
    x_{j}^{SKPR} &= 0 \quad \forall j \in \{ \mathcal{J}_{q}/\tilde{S}_{q} \}.
\end{align*}
\]

Note that by the construction of \((x_{q}^{SKPR}, s_{q}^{SKPR})\),

\[
f_{q} s_{q}^{SKPR} + \sum_{j \in \mathcal{J}_{q}} a_{j} x_{j}^{SKPR} = w_{q}^{KPEIR} \quad (q \in \mathcal{Q}).
\]
Since $\sum_{q \in Q} w_{q}^{KPEIR} \leq b$, clearly
\[ \sum_{q \in Q} f_{q} s_{q}^{SKPR} + \sum_{j \in J} a_{j} x_{j}^{SKPR} \leq b. \]

In addition,
\[ s_{q}^{SKPR} \geq x_{j}^{SKPR} \quad j \in J; \quad q \in Q \]
and
\[ x_{j}^{SKPR} \in [0, 1] \quad j \in J. \]

Therefore, the feasible solution ($w^{KPEIR}$) to KPEIR equates to a feasible solution to SKPR. Likewise, it is easy to see that a feasible solution to SKPR can be constructed from a feasible solution to SKPR by setting the flexibility variable, $w_{q}^{KPEIR}$, equal to the total capacity consumed by customers of type $q$ in the optimal solution solution ($x^{SKPR}, s^{SKPR}$).

Lastly, by the manner in which the function $\theta_{q} (q \in Q)$ is defined above, a solution to KPEIR, with objective function $z^{KPEIR}$ can be converted to a solution with an equivalent objective $z^{SKPR}$ and vice versa.

The equivalence of SKPR and KPEIR, along with the structural property in Theorem 9 reveal a great deal about the structure of SKPR. Before continuing, it will be useful to introduce the following sets. Let

\[ P = \{ q : 0 < s_{q} < 1 \} \]
be the set of partially included customer types. Furthermore, let
\[ F_{q} = \{ j \in J_{q} : 0 < x_{j} < 1 \} \]
be the set of customers of type $q$ that are fractionally assigned and let
\[ S = \{ q : |F_{q}| \geq 1 \} \]
be the set of customer types to which the fractional customers belong.
The following lemma bounds the number of fractional shared resource consumption variables $s_{q}^{SKPR}$ ($q \in Q$).

**Lemma 9.** An optimal solution to SKPR ($x^{SKPR}, s^{SKPR}$) exists for which

$$|\mathcal{P}| \leq 1.$$  

**Proof.** From the proof of Theorem 9, a fractional $s_{q}$ ($q \in Q$) only occurs when

$$0 < w_{q}^{KPEIR} < \tilde{a}_{k_{q}}.$$  

(7–28)

However, $w_{q}^{KPEIR}$ in the interval specified by (7–28) satisfies one of the two properties stated in Lemma 8. Thus, there is at most one $q$ such that $0 < w_{q}^{KPEIR} < \tilde{a}_{k_{q}}$ and the result immediately follows.

Lemma 10 provides additional information regarding the levels at which fractionally assigned customers of a specific type are assigned, as well as the number of customer types to which fractionally assigned customers belong.

**Lemma 10.** There exists an optimal solution to SKPR, ($x^{SKPR}, s^{SKPR}$) for which

(i) there is at most one customer type, say $\bar{q}$, which consists of fractional customers (i.e. $|\mathcal{S}| \leq 1$);

(ii) at least $|\mathcal{F}_{\bar{q}}| - 1$ fractional customers of type $\bar{q}$ are assigned at the same level $x_{j}^{SKPR} = x_{j'}^{SKPR}$ ($j, j' \in \mathcal{F}_{\bar{q}}$);

**Proof.** From the proof of Theorem 9, a fractional $s_{q}^{SKPR}$ or $x_{j}^{SKPR}$ ($j \in \mathcal{J}_{q}; q \in Q$) exists only if the value of $w_{q}^{KPEIR}$ is between two breakpoints of $\theta_{q}$ ($q \in Q$). Of course, since $\theta_{q}(w_{q})$ ($q \in Q$) is comprised of individual non-increasing linear segments, it is easy to see that there is at most one customer type for which $w_{q}^{KPEIR}$ falls strictly between the end points of a single linear segment. If this occurs such that

$$\tilde{a}_{k_{q}} < w_{q}^{KPEIR} < \tilde{a}_{k_{q}+1}$$

149
for some \( k_q \geq \hat{k}_q \), then this corresponds to a solution in which a single customer, \( \hat{j}_{k_q} \), is fractionally included. In addition, from the proof of Theorem 9, if this occurs with

\[
0 < u_q^{KPEIR} < \tilde{a}_{\alpha_{q_i}},
\]

this coincides with a solution in which \( s_q^{SKPR} \) is fractional and all partially assigned customers of type \( q \), \( \tilde{j}_q \in \mathcal{F}_q \) are assigned at the level \( s_q^{SKPR} \times \frac{\tilde{a}_{\alpha}}{\alpha_{q_i}} \). Thus, the second result follows from the fact that \( \tilde{a}_{\alpha_{j_q}} \neq a_{j_q} \) for at most one customer \( \tilde{j}_q \in \mathcal{F}_q \).

The result of Lemma 10 can be used to develop various heuristic rounding strategies. If the optimal solution to SKPR is indeed fractional, then we know that the fractional variables are limited to a single customer type. We use the results of this section to generalize our results to the flexible variant of our problem, FASR. We first determine the appropriate alternative formulation of FASR to consider and then study the resulting class of subproblems that are closely related to the problems studied in this section.

### 7.5 Flexible Customer Demand Generalization

In this section we consider an exact algorithm for the generalization of RCAS that allows the decision maker to determine the level at which each customer’s demand is fulfilled. Recall, from Section 7.2, that we refer to this model as FASR. Clearly, the exact algorithm proposed in Section 4.1 is not directly applicable to FASR. We consider an alternative formulation of FASR in which the decision variables represent the inclusion of a specific feasible subset of customers. Each subset of customers is accompanied by one of a finite number of demand fulfillment solutions, each specifically characterized in our presentation of the problem to follow. In this case, the so-called master problem that equivalently represents FASR is presented with the following definitions and notation.

- A so-called column in MP-F consists of a pair of vectors \((x_i, v_i)\). That is, \((x_i^d, v_i^d)\) is the \(d\)th vector pair (column), with binary elements \(x_{ij}^d (j \in \mathcal{J})\) indicative of a subset of customers that satisfies facility \(i\)’s capacity constraint; i.e.

\[
\sum_{q \in \mathcal{Q}} f_{iq} \max \{x_{ij}^d\} + \sum_{j \in \mathcal{J}} \ell_{ij} x_{ij}^d \leq b_i.
\] (7–29)

150
Furthermore, the fulfillment levels $v_{ij}^d$ ($j \in J$) correspond to an extreme point solution to the following linear program, solved with respect to the assignments $x_{ij}^d$,

$$\text{maximize } \sum_{j \in J} r_{ij} v_{ij} + C_{ij}^d$$

subject to

\begin{align*}
\sum_{j \in J} v_{ij} &\leq \bar{b}_{ij}^d \quad (7-30) \\
\ell_{ij} x_{ij}^d &\leq v_{ij} \leq u_{ij} x_{ij}^d \quad j \in J \quad (7-31)
\end{align*}

where $C_{ij}^d = \sum_{j \in J} p_{ij} x_{ij}^d$ and $\bar{b}_{ij}^d = b_{ij} - \sum_{q \in Q} f_q \max_{j \in J_q} \{ x_{ij}^d \}$;

- $S_i$ is the number of unique subsets of customers that can be assigned to facility $i$ while satisfying (7–29);
- $D_i$ is the set of indices, $d$, for all columns (as characterized in the first bullet) associated with facility $i$;
- $D_i$ is partitioned into sets $D_{is}$ such that $x_{ij}^d = x_{ij}^{d'}$ whenever $d, d' \in D_{is}$ and $x_{ij}^d \neq x_{ij}^{d'}$ whenever $d \in D_{is}$ and $d' \in D_{is'}$ with $s \neq s'$;
- $\gamma_{si}^d$ is a binary variable with value 1 if the $s$th subset of assignments associated with facility $i$ is used and 0 otherwise;
- $\lambda_i^d$ is a continuous variable with value in $[0, 1]$ representing the proportion of the values $(x_{ij}^d, v_{ij}^d)$ included in the solution to MP-F;

- $\alpha_i(x_{ij}^d, v_{ij}^d)$ is $\sum_{j \in J} (p_{ij} x_{ij}^d + r_{ij} v_{ij}^d)$;
- $\beta_i^d(x_{ij}^d, v_{ij}^d) = \sum_{j \in J_q} v_{ij}^d$.

Formally, the master problem, MP-F, is written as follows

$$\text{maximize } \sum_{i \in I} \sum_{d \in D_i} \alpha_i(x_{ij}^d, v_{ij}^d) \lambda_i^d$$

subject to

\begin{align*}
\sum_{i \in I} \sum_{d \in D_i} \beta_i^d(x_{ij}^d, v_{ij}^d) \lambda_i^d &\leq g_q \quad q \in Q \quad (7-32) \\
\sum_{i \in I} \sum_{d \in D_i} x_{ij}^d \lambda_i^d &= 1 \quad j \in J \quad (7-33)
\end{align*}
\[ \sum_{d \in D_i} \lambda_i^d = \gamma_i^s \quad s = 1, \ldots, S_i; \quad i \in I \quad (7-34) \]
\[ \sum_{s \in S_i} \gamma_i^s = 1 \quad i \in I \quad (7-35) \]
\[ \lambda_i^d \geq 0 \quad d \in D_i; \quad i \in I \quad (7-36) \]
\[ \gamma_i^s \in \{0, 1\} \quad s = 1, \ldots, S_i; \quad i \in I. \quad (7-37) \]

The following theorem ensures that MP-F is a valid representation of FASR.

**Theorem 10.** In the optimization problem MP-F,

(i) The number of columns, \((x_i^d, v_i^d), (d \in D_i)\) is finite;

(ii) MP-F and FASR are equivalent.

**Proof.** For (i), note that the dimension of the feasible subsets of customers associated with any facility is clearly exponential. Furthermore, recall that MP-F only considers columns in which the demand fulfillment levels for a fixed vector of assignments, \(x_i^d\), correspond to an extreme point solution to SP-v_i(x_i^d). Since the number of extreme points in the feasible region of SP-v_i is finite, the number of columns in MP-F that correspond to the same set of customer assignments to a particular facility is finite as well. Thus, since the number of facilities in MP-F is fixed, the number of columns included in MP-F is finite.

For (ii): Let \((\bar{\lambda}, \bar{\delta})\) be a solution to MP-F. Furthermore, let \((x^F, v^F)\) be a solution in terms of decision variables in FASR. First, if we set

\[ x^F_{ij} = \sum_{d \in D_i} \bar{\lambda}^d_i x^d_{ij} \quad i \in I; \quad j \in J \quad (7-38) \]

then by (7-33)–(7-37),

\[ \sum_{i \in I} x^F_{ij} = 1 \quad i \in I; \quad j \in J, \quad (7-39) \]

satisfying assignment constraint (7-3). Now set

\[ v^F_{ij} = \sum_{d \in D_i} \bar{\lambda}^d_i v^d_{ij} \quad i \in I; \quad j \in J. \quad (7-40) \]
Recall that, by definition, \((x^d_i, v^d_i)\) is (i) feasible to both facility capacity constraint (7–1) and flexibility constraint (7–4) and (ii) \(v^d_i\) is an extreme point solution to the feasible region of \(\text{SP-}v_i(x^d_i)\). Since \(v^F_{ij}\) is represented as a convex combination of these extreme points, \((x^F, v^F)\) is feasible to (7–1) and (7–4), as well. Moreover, by (7–32) we have that

\[
\sum_{j \in J} v^F_{ij} \leq g_q \quad \text{for } i \in I; q \in Q, \tag{7–41}
\]

satisfying customer type capacity constraint (7–2). Lastly,

\[
\sum_{i \in I} \sum_{d \in D_i} \alpha_i(x^d_i, v^d_i) \tilde{\lambda}^d_i = \sum_{i \in I} \sum_{d \in D_i} \left( \sum_{j \in J} (p_{ij}x^d_{ij} + r_{ij}v^d_{ij}) \right) \tilde{\lambda}^d_i \tag{7–42}
\]

\[
= \sum_{i \in I} \sum_{j \in J} \left( p_{ij} \sum_{d \in D_i} \tilde{\lambda}^d_i x^d_{ij} + r_{ij} \sum_{d \in D_i} \tilde{\lambda}^d_i v^d_{ij} \right) \tag{7–43}
\]

\[
= \sum_{i \in I} \sum_{j \in J} \left( p_{ij}x^F_{ij} + r_{ij}v^F_{ij} \right) \tag{7–44}
\]

so the objective value associated with \((\tilde{\lambda}, \tilde{\delta})\) in MP-F is equivalent to that of the constructed solution \((x^F, v^F)\) in FASR. Thus, any feasible solution to MP-F is also feasible to FASR and their objective function values are equivalent.

To show formulation equivalence, we must also be certain that any feasible solution to FASR is also feasible to MP-F, again with equivalent objective values. However, given any feasible solution \((x^F, v^F)\), the definition of the columns included in MP-F ensures that we can represent this solution as a convex combination of the columns comprising MP-F by choosing the appropriate values for \(\tilde{\lambda}^d_i\) associated with columns corresponding to the customer subsets indicated by \(x^F\). This immediately ensures that (7–33)–(7–37) are satisfied. Furthermore, since \((x^F, v^F)\) is feasible to (7–2), the customer type capacity constraint (7–32) is satisfied as well. Lastly, by a reverse presentation of (7–42)–(7–44), the objective values of the solutions to each formulation are equivalent. This shows the desired result.
In our exact algorithm, we solve the linear relaxation of MP-F (which we denote as LP(MP-F)), at each node of a branch-and-bound tree. However, as described in part (i) of the proof of Theorem 10, the total number of columns in MP-F is exponential. Therefore, we solve LP(MP-F) by adding columns iteratively to a restricted version of MP-F in which a subset of columns is considered. We denote this restricted relaxation as LP(RMP-F).

It is important to note that part (i) of Theorem (10) ensures that this column generation procedure has finite convergence. The derivation of the so-called pricing problem, solved to identify attractive columns, requires a study of the dual of LP(RMP-F). However, to simplify the presentation of this relaxation and its corresponding dual, first notice that MP-F can be equivalently reformulated as

\[
\text{maximize } \sum_{i \in I} \sum_{d \in D_i} \alpha_i(x^d_i, v^d_i) \lambda^d_i \\
\text{subject to} \quad \sum_{i \in I} \sum_{d \in D_i} \beta_i^q(x^d_i, v^d_i) \lambda^d_i \leq g_q \quad q \in Q \\
\sum_{i \in I} \sum_{d \in D_i} x^d_{ij} \lambda^d_i = 1 \quad j \in J \\
\sum_{d \in D_i} \lambda^d_i = 1 \quad i \in I \quad (7-45) \\
\lambda^d_i \geq 0 \quad d \in D_i; \ i \in I \quad (7-46) \\
\sum_{d \in D_{is}} \lambda^d_i \in \{0, 1\} \quad s = 1, \ldots, S_i; \ i \in I \quad (7-47)
\]

by substituting (7–34) into (7–35) and replacing (7–37) with (7–47). Therefore, the relaxed version of MP-F, which includes only a restricted set columns, LP(RMP-F), can be written as

\[
\text{maximize } \sum_{i \in I} \sum_{d \in D_i} \alpha_i(x^d_i, v^d_i) \lambda^d_i
\]
subject to (LP(RMP-F))

\[
\sum_{i \in I} \sum_{d \in D_i} \beta_i^q(x_i^d, v_i^d) \lambda_i^d \leq g_q \quad q \in Q \tag{7-48}
\]

\[
\sum_{i \in I} \sum_{d \in D_i} x_{ij}^d \lambda_i^d = 1 \quad j \in J \tag{7-49}
\]

\[
\sum_{d \in D_i} \lambda_i^d = 1 \quad i \in I \tag{7-50}
\]

\[
\lambda_i^d \geq 0 \quad d \in D_i; \ i \in I \tag{7-51}
\]

\[
\sum_{d \in D_{is}} \lambda_i^d \leq 1 \quad s = 1, \ldots, S_i; \ i \in I \tag{7-52}
\]

Notice that in LP(RMP-F), the binary restriction (7–47) is relaxed to (7–52) and the sets \( \tilde{D}_i, \tilde{D}_{is} \) and \( \tilde{S}_i \) contain the indices of the subset of columns being considered at any particular iteration of our column generation procedure. Furthermore, note that because of (7–50), constraint (7–52) is redundant in LP(RMP-F). Thus, the dual of LP(RMP-F) is correctly defined with respect to constraints (7–48)–(7–51) only. The purpose of the pricing problem is to identify a violated constraint in the following optimization problem, D(RMP-F),

\[
\text{minimize} \sum_{q \in Q} g_q \mu_q + \sum_{j \in J} \pi_j + \sum_{i \in I} \delta_i
\]

subject to (D(RMP-F))

\[
\sum_{q \in Q} \beta_i^q(x_i^d, v_i^d) \mu_q + \sum_{j \in J} x_{ij}^d \pi_j + \delta_i \geq \alpha_i(x_i^d, v_i^d) \quad d \in D_i; \ i \in I \tag{7-53}
\]

\[
\mu_q \geq 0 \quad q \in Q \tag{7-54}
\]

\[
\pi_j \text{ free} \quad j \in J \tag{7-55}
\]

\[
\delta_i \text{ free} \quad i \in I \tag{7-56}
\]

where
\[ \mu_q \ (q \in Q) \] are the dual variables associated with the customers set capacity constraint, (7–48) in LP(RMP-F);

\[ \pi_j \ (j \in J) \] are the dual variables associated with constraints, (7–49) in (LP(RMP-F));

\[ \delta_i \ (i \in I) \] are the dual variables associated with the convexity constraints, (7–50), in LP(RMP-F).

If we substitute the definitions of \[ \beta_q \] and \[ \alpha_q \] into this formulation,

\[
D(RMP-F) \text{ is equivalently represented by}
\[
\begin{align*}
\text{minimize} & \sum_{q \in Q} g_q \mu_q + \sum_{j \in J} \pi_j + \sum_{i \in I} \delta_i \\
\text{subject to} & (D(RMP-F)) \\
& \sum_{q \in Q} \sum_{j \in J} q v_{ij} - \sum_{j \in J} \sum_{q \in Q} \mu_q v_{ij} - \delta_i \geq \sum_{j \in J} (p_{ij} x_{ij}^d + r_{ij} v_{ij}^d) \quad d \in D_i; \ i \in I \\
& \mu_q \geq 0 \quad q \in Q \\
& \pi_j \text{ free} \quad j \in J \\
& \delta_i \text{ free} \quad i \in I
\end{align*}
\]

(7–57)

Our pricing problem seeks to identify a pair of vectors \((x_i, v_i)\), which violates (7–57) with (i) the vector \(x_i\) satisfying (7–29) and (ii) the corresponding demand fulfillment levels \(v_i\) determined by (SP-\(v_i\)). A vector \((x_i, v_i)\) violates (7–57) if

\[
\sum_{j \in J} [(p_{ij} - \pi_j) x_{ij} + r_{ij} v_{ij}] - \sum_{q \in Q} \sum_{j \in J} \mu_q v_{ij} - \delta_i > 0
\]

(7–61)

or equivalently

\[
\sum_{j \in J} [(p_{ij} - \pi_j) x_{ij} + (r_{ij} - \mu_q_j) v_{ij}] - \delta_i > 0
\]

(7–62)

where \(q_j\) is the type to which customer \(j\) belongs. Therefore, in light of (i) and (ii), along with dual constraint (7–57), the pricing problem associated with facility \(i \ (i \in I)\) is given by

\[
\text{maximize} \sum_{j \in J} [(p_{ij} - \pi_j) x_j + (r_{ij} - \mu_q_j) v_j] - \delta_i
\]
subject to (PP_{i-F})

$$\sum_{q \in Q} f_{iq} \max \{x_j\} + \sum_{j \in J} v_j \leq b_i$$

$$\ell_{ij} x_j \leq v_j \leq u_{ij} x_j \quad j \in J$$

$$x_j \in \{0, 1\} \quad j \in J.$$  

In the following section we study the class of problems that includes PP_{i-F} by extending the results of Section 7.4.

### 7.6 Shared Consumption Knapsack problem with Flexible Customer Demand

In this section, we study the following class of knapsack problems.

$$\text{maximize} \sum_{j \in J} (p_j x_j + r_j v_j)$$

subject to (SKFP)

$$\sum_{q \in Q} f_q s_q + \sum_{j \in J} v_j \leq b$$

$$v_j \leq u_j x_j \quad j \in J \quad (7-63)$$

$$v_j \geq \ell_j x_j \quad j \in J \quad (7-64)$$

$$s_q \geq x_j \quad j \in J_q; q \in Q$$

$$x_j \in \{0, 1\} \quad j \in J.$$  

$$s_q \in \{0, 1\} \quad q \in Q.$$  

whose linear relaxation is given by

$$\text{maximize} \sum_{j \in J} (p_j x_j + r_j v_j)$$

subject to (SKFPR)

$$\sum_{q \in Q} f_q s_q + \sum_{j \in J} v_j \leq b$$
\[ v_j \leq u_j x_j \quad j \in \mathcal{J} \]
\[ v_j \geq \ell_j x_j \quad j \in \mathcal{J} \]
\[ s_q \geq x_j \quad j \in \mathcal{J}_q; \quad q \in \mathcal{Q} \]
\[ x_j \in [0, 1] \quad j \in \mathcal{J}. \]

As in the SKP', the non-linear representation of SKFP, SKFP', is given by

\[
\text{maximize } \sum_{j \in \mathcal{J}} (p_j x_j + r_j \nu_j) \\
\text{subject to } (\text{SKFP'})
\]

\[
\sum_{q \in \mathcal{Q}} \left( f_q + \sum_{j \in \mathcal{J}_q} \nu_j \right) s_q \leq b \\
\nu_j \leq u_j y_j \quad j \in \mathcal{J} \\
\nu_j \geq \ell_j y_j \quad j \in \mathcal{J} \\
 s_q \geq y_j \quad j \in \mathcal{J}_q; \quad q \in \mathcal{Q} \\
y_j \in \{0, 1\} \quad j \in \mathcal{J} \quad (7–65) \\
s_q \in \{0, 1\} \quad q \in \mathcal{Q}
\]

where \( \nu_j \ (j \in \mathcal{J}_q) \) are used to represent the demand fulfillment levels and \( y_j \ (j \in \mathcal{J}_q) \) the assignments in SKFP'. As with SKP', we consider the following relaxation of SKFP'

\[
\text{maximize } \sum_{j \in \mathcal{J}} (p_j y_j + r_j \nu_j) \\
\text{subject to } (\text{SKFPR'})
\]

\[
\sum_{q \in \mathcal{Q}} \left( f_q + \sum_{j \in \mathcal{J}_q} \nu_j \right) s_q \leq b \\
\nu_j \leq u_j y_j \quad j \in \mathcal{J}
\]
\[ \nu_j \geq \ell_j y_j \quad j \in \mathcal{J} \]
\[ s_q \geq y_j \quad j \in \mathcal{J}_q; \ q \in \mathcal{Q} \]
\[ y_j \in [0, 1] \quad j \in \mathcal{J} \]
\[ s_q \in \{0, 1\} \quad q \in \mathcal{Q} . \]

in which the binary restrictions (7–65) are relaxed. Interestingly, SKFPR' can also be reformulated as the KPEI presented in Section 7.4 with \( \ell'_q = \min\{f_q, b\}, u'_q = \min\left\{\left(f_q + \sum_{j \in \mathcal{J}_q} u_j\right), b\right\} \) \( (q \in \mathcal{Q}) \) and \( \tilde{r}_q(w_q) \) the optimal solution to the following parametric optimization problem, FCKP,

\[
\begin{align*}
\text{maximize} & \quad \sum_{j \in \mathcal{J}} p_j y_j + r_j \nu_j \\
\text{subject to} & \quad f_q + \sum_{j \in \mathcal{J}_q} \nu_j \leq w_q \\
& \quad \ell_j y_j \leq \nu_j \leq u_j y_j \quad j \in \mathcal{J}_q \\
& \quad y_j \in [0, 1] \quad j \in \mathcal{J}_q .
\end{align*}
\]

To more explicitly state \( \tilde{r}_q(w_q) \), we define the following sets and notation. Let \( \mathcal{J}_q^- = \{ j \in \mathcal{J}_q : p_j \leq 0 \} \) be the set of customers with a non-positive fixed profit and \( \mathcal{J}_q^+ = \{ j \in \mathcal{J}_q : p_j > 0 \} \) the customers with a positive fixed profit. Furthermore, split each \( j \in \mathcal{J}_q^+ \) into two customers, \( j_1 \) and \( j_2 \). The following lemma characterizes \( \tilde{r}_q(w_q) \) \( (q \in \mathcal{Q}) \) using the definition provided in Lemma 6.

**Lemma 11.** For SKFPR', the corresponding function \( \tilde{r}_q \) \( (q \in \mathcal{Q}) \) is given by Lemma 6 with \( \mathcal{J}_q = \{ \mathcal{J}_q^- \cup \mathcal{J}_q^+ \} \) used in place of \( \mathcal{J}_q \) and

\[
\begin{align*}
a_{j_1} &= \ell_j \quad (7–66) \\
a_{j_2} &= u_j - \ell_j \quad (7–67)
\end{align*}
\]
\[ p_{j_1} = p_j + r_j \ell_j \] (7–68)

\[ p_{j_2} = r_j (u_j - \ell_j) \] (7–69)

for \( j_1 \) and \( j_2 \) in \( J_q^+ \) and

\[ a_j = u_j \] (7–70)

\[ p_j = p_j + r_j u_j \] (7–71)

for \( j \in J_q^- \).

**Proof.** Notice that the FCKP is itself the KPEI with a linear objective. Section 4.2.2 of Chapter 4 provides a separate algorithm for this problem. This algorithm splits each customer \( j \in J^+ \) into two parts. The first part has customer demand size \( \nu_{j_1} \in [0, \ell_j] \) and a per unit profit given by \( \sigma_{j_1} = \frac{p_j}{\ell_j} + r_j \). The second part has customer demand size \( \nu_{j_2} \in [0, u_j - \ell_j] \) and a per unit profit \( \sigma_{j_2} = r_j \). Customers \( j \in J^- \) have customer demand size \( \nu_j \in [0, u_j] \) and a per unit profit given by \( \sigma_j = \frac{p_j}{u_j} + r_j \). In Chapter 4 we showed that the linear version of FCKP can be reformulated as a CKP with customers \( j \in J_q^- \) and \( j_1 \) and \( j_2 \) in \( J_q^+ \). As established in Section 7.4, the definition of \( \tilde{r}_q (q \in Q) \) in Lemma 6 contains a solution to the CKP. Therefore, by replacing \( J_q \) with a set \( \hat{J}_q \) that includes all customers in both \( J_q^- \) and \( J_q^+ \) with the specified profit and resource consumption parameters, Lemma 6 defines \( \tilde{r}_q (q \in Q) \) with the additional consideration of flexible demand.

Our approaches will again be motivated by KPEIR presented in the previous section. Here again, the function \( \theta_q (q \in Q) \) is a non-decreasing concave envelope encompassing the origin, the function \( \tilde{r}_q \) described in Lemma 11 and the point \((b, \tilde{r}_q(u'_q))\). Similar to the result of Lemma 11, \( \theta_q(w_q) (q \in Q) \), for the flexible variant of the problem, is defined by Lemma 7 with \( \hat{J}_q \) used in lieu of \( J_q \) and parameters for each “customer” \( j \in \hat{J}_q \) given by (7–66)–(7–71). Therefore, the algorithm presented in the previous section still applies.
using the parameters given by (7–66)–(7–71) to define \( \hat{p} \) and \( \hat{a} \) for the expanded customer set.

Interestingly, there is a relationship between KPEIR and SKFPR which is given in the following theorem. Note that \( \hat{J}_q \) may consist of customers in \( J_q^- \), or \( J_q^+ \), or both. That is, up to two “customers” in \( \hat{J}_q \) may be associated with the same true customer in SKFPR. Therefore, let \( \bar{j}_{kq} \) be the customer in SKFPR to which an individual “customer” \( \hat{j}_{kq} \in \hat{J}_q \) corresponds. Furthermore, let \( x^t \) be a vector of temporary assignments used to simplify the representation of the following theorem. Using this notation, we state the relationship between optimal solutions to KPEIR and SKFPR.

**Theorem 11.** Given an optimal solution to KPEIR \((w^{KPEIR})\) with \( \bar{r}_q(w_q) \) defined by Lemma 11, an optimal solution to SKFPR \((x^{SKFPR}, v^{SKFPR}, s^{SKFPR})\) is given by the following.

If \( w_q^{KPEIR} = 0 \),

\[
\begin{align*}
S_q^{SKPR} &= 0 \\
x_j^{SKPR} &= 0 \\
v_j^{SKPR} &= 0
\end{align*}
\]

\( j \in J_q \).

If \( 0 < w_q^{KPEIR} < \tilde{a}_{kq} \), let \( \tilde{x} = \frac{w_q^{KPEIR}}{\tilde{a}_{kq}} \), then

\[
\begin{align*}
S_q^{SKPR} &= \tilde{x} \\
x_{jkq}^t &= \tilde{x} \left( \frac{\tilde{a}_{jkq}}{a_{jkq}} \right) \\
x_{jkq}^t &= 0 \\
v_{jkq}^t &= 0
\end{align*}
\]

\( k_q = 0, \ldots, k_q \)

\( \forall j_{kq} \); \( k_q = k_q + 1, \ldots, |\bar{S}_q| - 1 \)

\( j \in \{\hat{J}_q/\bar{S}_q\} \)

\[
\begin{align*}
x_j^{SKPR} &= \max_{j_{kq} \bar{j}_{kq} = j} \left\{ x_{jkq}^t \right\} \\
v_j^{SKPR} &= x_j^{SKPR} u_j \\
v_j^{SKPR} &= x_{jkq}^t \ell_j + x_{jkq}^t (u_j - \ell_j) \\
&= (k_q', k_q'') \ni \bar{j}_{kq}' = \bar{j}_{kq}'' = j \text{ and } k_q' < k_q'' \ni j \in \{J_q^+ \cap \bar{S}_q\}
\end{align*}
\]
\[ v_j^{SKFPR} = 0 \quad j: \bar{j}_k \neq j \forall k_q = 0, \ldots, |S_q| - 1; \ q \in Q. \]

Lastly, if \( \bar{a}_k \leq w_q^{KPEIR} < \bar{a}_{k+1} \) for some \( \hat{k}_q \geq k_q \),

\[ s_q^{SKFPR} = 1 \]

\[ x^t_{j_{k_q}} = 1 \quad k_q = 0, \ldots, \hat{k}_q \]

\[ x^t_{j_{k_q+1}} = \frac{w_q^{KPEIR} - \bar{a}_{\hat{k}_q}}{a_{j_{k_q+1}}} \]

\[ x^t_{j_{\hat{k}_q}} = 0 \quad k_q = \hat{k}_q + 2, \ldots, |\tilde{S}_q| - 1 \]

\[ x^t_j = 0 \quad j \in \{ \hat{J}_q / S_q \} \]

\[ x_j^{SKFPR} = \max_{j_{k_q} \mid j_{k_q} = j} \{ x^t_{j_{k_q}} \} \quad j \in J_q \]

\[ v_j^{SKFPR} = x_j^{SKFPR} u_j \quad j \in \{ J^- \cap \tilde{S}_q \} \]

\[ v_j^{SKFPR} = x^t_{j_{k'_q}} \ell_j + x^t_{j_{k''_q}} (u_j - \ell_j) \quad \left( k'_q, k''_q \right): \bar{j}_{k'_q} = \bar{j}_{k''_q} = j \text{ and } k'_q < k''_q; \ j \in \{ J^+ \cap \tilde{S}_q \} \]

\[ v_j^{SKFPR} = 0 \quad j: \bar{j}_k \neq j \forall k_q = 0, \ldots, |\tilde{S}_q| - 1; \ q \in Q. \]

Proof. By the construction of \( (x^{SKFPR}, v^{SKFPR}, s^{SKFPR}) \),

\[ f_q s_q^{SKFPR} + \sum_{j \in J_q} v_j^{SKFPR} = w_q^{KPEIR} \quad (q \in Q). \]

Since \( \sum_{q \in Q} w_q^{KPEIR} \leq b \), clearly

\[ \sum_{q \in Q} f_q s_q^{SKFPR} + \sum_{j \in J_q} v_j^{SKFPR} \leq b. \]

Also,

\[ s_q^{SKFPR} \geq x_j^{SKFPR} \quad j \in J_q; \ q \in Q \]

and

\[ x_j^{SKFPR} \in [0, 1] \quad j \in J \]

and

\[ x_j^{SKFPR} \ell_j \leq v_j^{SKFPR} \leq x_j^{SKFPR} u_j \quad j \in J. \]
Therefore, the feasible solution \( w^{KPEIR} \) to KPEIR equates to a feasible solution to SKPR. Furthermore, a feasible solution to KPEIR can be constructed from a feasible solution to SKFPR by setting the flexibility variable, \( w_q^{KPEIR} \), equal to the total capacity consumed by customers of type \( q \) in the optimal solution \((x^{SKFPR}, v^{SKFPR}, s^{SKFPR})\). As in Theorem 9, by the manner in which the function \( \theta_q \) \((q \in Q)\) is defined, a solution to KPEIR, with objective function \( z^{KPEIR} \), can be represented as a solution to SKFPR with an equivalent objective \( z^{SKFPR} \) and vice versa.

From the equivalence of KPEIR and SKFPR and Lemma 8, a great deal of information regarding the structure of SKFPR can be obtained. First note that a fractional \( s_q^{SKFPR} \) corresponds to a solution to KPEIR with \( 0 < w_q^{KPEIR} < \tilde{a}_{\bar{q}} \). By Lemma 10, this occurs for only one customer type. Therefore, Lemma 9 holds for SKFPR and there exists an optimal solution to SKFPR with at most one fractional shared resource consumption variable. Furthermore, the results of Theorem 10 can be extended to SKFPR with an additional result specifying the number of customers in which the corresponding fulfillment level is between its bounds; i.e. \( \ell_j < v_j^{SKFPR} < u_j \).

**Lemma 12.** There exists an optimal solution to SKFPR, \((x^{SKFPR}, s^{SKFPR}, s^{SKFPR})\) with the following properties.

(1) There is at most one customer type, say \( \bar{q} \) for which

(a) fractional customers exist (i.e. \( |S| \leq 1 \)), or

(b) a single customer is assigned between its bounds (i.e. \( \ell_j < v_j^{SKFPR} < u_j \)).

(2) At least \( |\mathcal{F}_q| - 1 \) fractional customers of type \( \bar{q} \) are assigned at the same level \( x_j^{SKPR} = x_{j'}^{SKPR} \) \((j, j') \in \mathcal{F}_q\).

**Proof.** Property 1(a) and 2 follow from the arguments offered in the proof of Lemma 10. Furthermore, a customer with Property 1(b) exists only if \( w_q^{KPEIR} \) is included strictly between two breakpoints of \( \theta_q(w_q) \) \((q \in Q)\). As stated in Lemma 10, this can only occur for a single customer. This yields the desired result. \( \square \)
Therefore, as with SKPR, an optimal solution to SKFPR can easily be converted to a feasible solution to SKFP. We will discuss specific ways to obtain a feasible solution from the optimal solution to SKFPR in Section 7.7.2. Furthermore, SKFPR can be solved at each node of a customized branch-and-bound procedure to solve SKFP to optimality.

7.7 Branch-and-Price Algorithm Implementation

Our implementation and computational testing will focus solely on the flexible variant of the problem, FASR, discussed in Section 7.5. The implementation of our branch-and-price algorithm is partially motivated by the demonstrated success of the choices made in Section 4.3. However, as a result of the unique characteristics of MP-F and the pricing problem PP\(_i\)-F, some additional considerations must be given to the implementation of the branch-and-price algorithm for the FASR.

7.7.1 Initial Feasible Solution

To ensure that a feasible solution exists to LP(RMP-F) we propose a slightly modified two-phase procedure from that used to solve LP(RSP) in Chapter 4. Phase 1 of our approach is used to generate a feasible set of columns to LP(RMP-F). To this end, we include (nonnegative) slack variables for each customer type capacity constraint (7–32) and assignment constraint (7–33). Our Phase 1 objective is then to minimize the sum of these slack variables. The resulting Phase 1 problem is thus given by

\[
\begin{align*}
\text{minimize} & \quad \sum_{j \in J} \varrho_j + \sum_{q \in Q} \varsigma_q \\
\text{subject to} & \quad (\text{RMP-F-Phase 1}) \\
& \quad \sum_{i \in I} \sum_{d \in D_i} \beta_i^d (x_{i}^d, v_i^d) \lambda_i^d - \varsigma_q \leq g_q \quad q \in Q \\
& \quad \sum_{i \in I} \sum_{d \in D_i} x_{ij}^d \lambda_i^d + \varrho_j = 1 \quad j \in J \\
& \quad \sum_{d \in D_i} \lambda_i^d = 1 \quad i \in I
\end{align*}
\]
\[ \lambda_i^d \geq 0 \quad \text{for} \quad d \in D_i; \quad i \in \mathcal{I} \]
\[ \sum_{d \in D_{is}} \lambda_i^d \in \{0, 1\} \quad \text{for} \quad s = 1, \ldots, S_i; \quad i \in \mathcal{I} \]

We solve the linear relaxation of (RMP-F-Phase 1) using column generation. The pricing problem is similar to (PP\textsubscript{i}-F), with a slightly modified objective.

\[
\text{maximize} \quad \sum_{j \in \mathcal{J}} (\pi_j x_j + \mu_q v_j) - \delta_i
\]

subject to

\[
\sum_{q \in \mathcal{Q}} s_q f_{iq} + \sum_{j \in \mathcal{J}} v_j \leq b_i
\]
\[
v_j \in [l_{ij}, u_{ij}] \quad \text{for} \quad j \in \mathcal{J}
\]
\[
s_q \geq x_j \quad \text{for} \quad j \in \mathcal{J}_q; \quad q \in \mathcal{Q}
\]
\[
x_j \in \{0, 1\} \quad \text{for} \quad j \in \mathcal{J}
\]
\[
s_q \in \{0, 1\} \quad \text{for} \quad q \in \mathcal{Q}.
\]

This problem is an SKFP with \( p_{ij} = 0 \) and \( r_{ij} = 0 \) (\( i \in \mathcal{I}; \quad j \in \mathcal{J} \)). Therefore, it can be solved as discussed in Section 7.6. If the optimal value of LP(RMP-F-Phase1) equals 0, any optimal solution to this problem is feasible for LP(RMP-F); otherwise, the problem instance is infeasible. In the former case we use this feasible solution to initialize the column generation procedure for solving LP(RMP-F).

### 7.7.2 Heuristics for PP\textsubscript{i}-F

As mentioned in Section 7.6, the result of Lemma 10 can be used to develop various heuristic rounding strategies. In our implementation, we consider two alternative strategies.

**Heuristic 1.** If the optimal solution to the linear relaxation of PP\textsubscript{i}-F, say LP(PP\textsubscript{i}-F), is indeed fractional, then we know that the fractional variables are limited to a single customer type. A rounding procedure similar to that commonly used for a knapsack
problem is as follows. Let $\bar{q} = \{q \in Q : |\mathcal{F}_q| \geq 1\}$ be the customer type associated with partially assigned customers in the solution to LP(PP$_i$-F). Set

$$x_j^{PP_i,F-H1} = x_j^{LP(PPI,F)}$$

$$v_j^{PP_i,F-H1} = v_j^{LP(PPI,F)}$$

$$j \in \{(J/\mathcal{F}_q)\}$$

and

$$s_q^{PP_i,F-H1} = \max_{j \in \mathcal{J}_q} \{x_j^{PP_i,F-H1}\}; \quad q \in Q.$$

In the above procedure, all fractional assignment variables and the corresponding demand fulfillment levels in the LP-relaxation of PP$_i$-F are set to zero.

The next heuristic uses the fractional variables to make assumptions about which customer types are included in the solution. Then, a secondary optimization problem is solved to determine the corresponding set of customer to include and the subsequent demand fulfillment levels.

**Heuristic 2.** Rather than simply removing all partially assigned customers, we can alternatively attempt to fully include a partial set of these fractional customers. To accomplish this, we use the solution to LP(PP$_i$-F) to determine which customer types are included in our heuristic solution. That is, set

$$s_q^{PP_i,F-H2} = \left\lceil s_q^{LP(PPI,F)} \right\rceil \quad q \in Q.$$

Then $\tilde{Q} = \{q \in Q : s_q^{PP_i,F-H2} = 1\}$ is the set of customer types included in the heuristic solution. Set

$$x_j^{PP_i,F-H2} = 0$$

$$v_j^{PP_i,F-H2} = 0$$

$$j \in \mathcal{J}_q; \quad q \in \{Q/\tilde{Q}\}.$$

Using the set of customer types, $\tilde{Q}$, we solve the following optimization problem

$$\text{maximize} \sum_{q \in \tilde{Q}} \sum_{j \in \mathcal{J}_q} \left[ (p_{ij} - \pi_j) x_j + (r_{ij} - \mu_{q_j}) v_j \right] - \delta_i$$
subject to \[(SP-H2)\]

$$
\sum_{q \in \tilde{Q}} \sum_{j \in J_q} v_j \leq b_i - \sum_{q \in \tilde{Q}} f_{iq} \\
\ell_j x_j \leq v_j \leq u_j x_j \quad j \in J_q; \ q \in \tilde{Q} \\
x_j \in [0, 1] \quad j \in J_q; \ q \in \tilde{Q}.
$$

SP-H2 is a KPEI'-R studied in Chapter 4 and can be solved very efficiently. Moreover, we know there exists an optimal solution to SP-H2 for which there is at most one fractionally assigned customer. Given the optimal solution to SP-H2, the remaining heuristic solution is given as follows. If one exists, let \(\bar{j}\) be the fractional customer included in the optimal solution \((x_{SP-H2}^{\text{opt}}, v_{SP-H2}^{\text{opt}})\). Set

$$
x_{j,PP,-F-H2} = x_{j,SP-H2}^{\text{opt}} \quad v_{j,PP,-F-H2} = v_{j,SP-H2}^{\text{opt}} \quad j \in \left( \bigcup_{q \in \tilde{Q}} J_q \right) / \bar{j} \\
x_{j,PP,-F-H2} = 0 \quad v_{j,PP,-F-H2} = 0.
$$

Our testing indicated that our heuristic procedures were most successful while solving LP(RMP-F) in the root node. At subsequent nodes of the tree we relied predominantly on the exact branch-and-bound algorithm to solve PP,\(F\). Note that Heuristic 2 clearly requires more computational effort than Heuristic 1. Therefore, our implementation only utilizes Heuristic 2 at the root node. That is, in the root node we first attempt to identify an attractive column using Heuristic 1. If we are unsuccessful, we consider the more intensive Heuristic 2. However, in non-root nodes, we only the solve the pricing problems heuristically via the rounding scheme of Heuristic 1. This implementation choice was shown to most consistently produce results in the least amount of time.

### 7.7.3 Solving LP(RMP-F)

Similar to our discussion of the branch-and-price algorithm presented in Chapter 4, at any node in our branch-and-bound tree we must solve a relaxation of (MP-F). Since our
pricing problem still decomposes by facility, there are again $|I|$ potential pricing problems to consider. Our rule for considering the various pricing problems is taken from Section 4.3.2 of Chapter 4. That is, at each iteration of our column generation procedure we solve all pricing problems heuristically according to the rules described in Section 7.7.2. All attractive columns are added to LP(RMP-F) and the column generation procedure continues. If no column is found to price out via our heuristic(s), we order the pricing problems in non-increasing order of the objective values determined by the heuristic. Pricing problems are solved via branch-and-bound until either a single column prices out, or it is determined that no column prices out. To reduce the effect of slow convergence as we approach the optimal solution to LP(RMP-F), we terminate our column generation procedure when our current LP(RMP-F) solution value is provably within $10^{-3}$ of the optimal solution to LP(RMP-F). Recall that the upper bound used to calculate this gap requires solving all pricing problems exactly. Therefore, we again only update the upper bound after solving $|I||J|$ pricing problems either heuristically or to optimality.

In Chapter 4, quality feasible solutions to the SP were obtained by solving RSP as an integer program using columns generated in solving LP(RSP) at the root node. We investigated this implementation choice for MP-F. Our testing showed that the time for CPLEX to solve this MIP was notably larger than in the case of SP. This is likely due to the alternative structure of MP-F versus SP, specifically, the addition of constraints (7–32). Moreover, we found that better solutions could be obtained at low levels of the tree. Therefore, unlike our implementation for SP, we do not solve an MIP using the columns found in the root node of the search tree.

7.7.4 Node and Variable Selection

Our node selection rule is motivated by the success of the implementation in Chapter 4. We initially search the tree using a depth-first rule. Once a feasible solution to MP-F is obtained, we explore the tree using a best-bound rule. This node selection policy is also
implemented in the branch-and-bound procedure used to solve our pricing problem to optimality.

Again, rather than branching on the \( \lambda \) values in LP(RMP-F) we branch on \( x \) variables that have a value of 1 in a column associated with a fractional \( \lambda \). The choice of \( x \) to branch on is based on the degree of fractionality of each variable in the solution to LP(RMP-F). We explored both least fractional (i.e., that variable which is closest to 0 or 1, where ties are broken arbitrarily) and most fractional (i.e., that variable which is closest to 0.5). While the difference in performance for the two approaches was very slight, the most fractional rule is used in the computational results shown in Section 7.8.

7.8 Computational study

In this section we provide a computational study of our branch-and-price algorithm. Since the motivation behind this chapter is the flexible variant of the problem, we focus on the more general FASR. Section 7.8.1 discusses the instance generation scheme chosen for this study. Then, in Sections 7.8.2 and 7.8.3 we discuss the performance of our algorithm versus the commercial solver CPLEX on a broad range of test instances.

7.8.1 Experimental Design

In our computational tests, our main set of instances considers 15 and 30 facilities with the number of customers equal to \( |J| = 2|I|, 3|I|, \) and \( 5|I| \). For each facility/customer combination we study instances with \( |Q| = 3 \) customer types. For each customer, we generate the random vectors of fixed profit parameters \( P_j \) and unit revenues \( R_j \) from uniform distributions on \([30, 50]\) and \([2, 5]\), respectively. Furthermore, the customer requirements \( L_j \) and \( D_j \) are generated from uniform distributions on \([75, 125]\) and \([15, 35]\), respectively. Here, \( L_j \) is a random vector of customer lower bounds and \( D_j \) is a random vector containing values representative of the difference between upper and lower bounds of a customer. We also generate shared capacity consumption parameters such that

\[
f_{iq} = \Theta \alpha_{iq}|J| \tag{7–72}
\]
where $\Theta$ is a non-negative parameter that measures the absolute magnitude of the shared resource consumption and $\alpha_{iq}$ is the relative magnitude of the shared resource consumption of type $q$ for facility $i$. Moreover,

$$\sum_{q \in Q} \alpha_{iq} = 1.$$  

For these experiments, we set $\alpha_{iq} = \frac{|J_q|}{|J|}$. In each of our tests the facility capacities are given by

$$b_i = (\beta + \Theta \phi_i) |J| \quad (i \in I). \quad (7-73)$$

where the parameter $\phi_i \in [0, 1]$ is a measure of the fraction of customer types that can be assigned to facility $i$ ($i \in I$) and again

$$\beta = \tau_a \cdot E \left( \min_{i \in I} (A_{i1} + L_{i1}) \right) \frac{|I|}{|J|} \quad (7-74)$$

The parameter $\tau_a$ measures the capacity available for variable consumption. In these tests we consider the flexibility level determined by setting $\tau_a = 1.2$. However, we consider alternative values of $\phi_i$. Lastly, our customer type capacity restrictions are given by

$$g_q = \tau_t \cdot E \left( \min_{i \in I} (A_{i1} + L_{i1}) \right) |J_q| \quad (q \in Q) \quad (7-75)$$

where $\tau_t$ determines the flexibility available for customers of type $q$ with respect to capacity $g_q$. In these tests we consider the flexibility level determined by setting $\tau_t = 1.2$, as well.

In our experimentation, we sought to compare the effectiveness of our branch-and-price approach against the commercial solver CPLEX. Each of our instances was run until either a solution value within .1% of the optimal solution was obtained or a time limit of one hour was reached. Our tables present results for 10 randomly generated instances for each combination of parameter settings. Specifically, each table reports

(i) the number of columns generated in the root node of the branch-and-price algorithm;
(ii) the total number of columns generated throughout the entire branch-and-price algorithm;

(iii) the number of nodes considered in the branch-and-price tree;

(iv) the amount of time required to solve the relaxed master problem at the root node;

(v) the total time required by the branch-and-price algorithm;

(vi) the total time required by the CPLEX, with the following additional information where appropriate:

- the superscript indicates the relative solution error calculated by using the solver’s best lower and upper bound, $z_{UB}^S$, $z_{LB}^S$; i.e.

\[
\text{error} = \frac{z_{UB}^S - z_{LB}^S}{z_{UB}^S} \times 100.
\]

All experiments were performed on a a PC with a 3.40 GHz Pentium IV processor and 2 GB of RAM. The mixed-integer programming problems as well as the relaxed master problems were solved using CPLEX 11.2. In Section 7.8.2 we discuss a base set of results for our branch-and-price algorithm. Then, in Section 7.8.3 we discuss how these results change with different instance generation parameters.

### 7.8.2 Base Results

Our main set of results considers two sets of instances: (i) with 15 facilities and number of customers equal to 30, 45 and 75 and (ii) with 30 facilities and 60, 90, and 150 customers. For both sets of instances, the customers are separated into three equal size sets (i.e. types). This main set of instances generates shared resource consumption variables (i.e. $f_{iq}$ ($i \in \mathcal{I}; \ q \in \mathcal{Q}$)) with magnitude parameter equal 5. The facility capacities are generated with $\phi_i = .5$. That is, we can expect 50% of the customer types to be able to be assigned to a particular facility. Lastly, as previously stated, the flexibility allowances are set at the moderate levels $\tau_a = 1.2$ and $\tau_t = 1.2$.

Tables 7-1 and 7-2 show that the branch-and-price algorithm solves the 15 facility instances with 30 and 60 customers in less time than CPLEX, on average. Each of the optimal solutions for the 15 facility/30 customer instances in Table 7-1 is obtained in
less than 100 nodes and the time to solve the root relaxation is less than 10 seconds. The average time for CPLEX to solve these instances is more than 6 times that of the branch-and-price algorithm. Interestingly, it is clear that a large portion of the columns considered in each of the instances were generated in the root node of the branch-and-price algorithm. Each of the 15 facility/45 customer instances in Table 7-2 was also solved in less time with the branch-and-price algorithm. However, the number of nodes considered and columns generated is substantially higher than that seen in 30 customer instances in Table 7-1. For the 45 customer instances, Table 7-2 shows that more than 1000 nodes were often required to obtain an optimal solution. Furthermore, a far higher percentage of columns was generated outside of the root node. While the computational requirements for the branch-and-price algorithm were notably increased in Table 7-2, it should be noted that CPLEX failed to solve 4 of the 10 instances within the allotted hour. Each of these 4 instances was solved to optimality via branch-and-price in less than 25 minutes. Unfortunately, neither CPLEX or our branch-and-price algorithm was able to solve instances with 15 facilities and 75 customers with the parameters specified at the beginning of this section within the 1 hour time limit.

Tables 7-3 and 7-4 consider problem instances with a larger number of facilities and customers. Specifically, 30 facility instances with 60 and 90 customers are considered. The same set of parameters is used to generate the demand requirements and capacity limitations. The performance of branch-and-price over CPLEX is even more clearly defined in these results. With 60 customers, Table 7-3 shows that branch-and-price takes an average of less than 30 seconds to solve these instances, while CPLEX does not solve any of the instances to the specified tolerance limits in the allotted time. Interestingly, even though the instances are much larger, the time to solve the root node problem is still less than 10 seconds. Similar to what was seen with the 15 facility instances, Table 7-4 shows again that the number of columns generated grows significantly as the customers per facility is increased to 3. However, the branch-and-price algorithm is still able to solve
all 10 instances in an average of less than 11 minutes, while again CPLEX fails to solve any of these instances within 1 hour.

Neither the branch-and-price algorithm, nor CPLEX is able to solve instances with 30 facilities/150 customers within an hour. Therefore, as is typical with branch-and-price approaches, our main results in Tables 7-1–7-4 suggest that our algorithm is most successful with a customer-to-facility ratio less than or equal to 3. Interestingly, while GAPFD and CFLFD (with linear revenue functions) could be solved efficiently by CPLEX for instances with customer-to-facility ratios of 5 or more, the same is clearly not true for FASR. This reinforces the difficulty of this class of problems. Of course, a number of different types of instances can be considered using the data model proposed in Section 7.8. In the following section, we provide a few insights into how instances with different characteristics may be easier or more difficult to solve than those considered in this section.

7.8.3 Extended Results

In this section, we consider various alternatives to the parameters used to generate instances in our main set of results. While it is impractical, of course, to consider all variations, this section strives to provide some insight into how alternative instances may impact the computational requirements of branch-and-price versus CPLEX. First, recall that the magnitude of the shared resource consumption (i.e. \( \Theta \)) was chosen to be 5 in our main results. In our computational study, we also considered magnitudes of 1 and 25 as well, with all remaining parameters the same. With \( \Theta = 25 \), Table 7-5 suggests the problems become dramatically easier for both branch-and-price and CPLEX. This is perhaps due to the fact that the shared resource consumption becomes the dominating component of the problem, with the flexible demand required by individual customers requiring less consideration. More likely, however, is that given the combination of extremely large magnitudes with a fairly large value \( \phi = 0.5 \), the facility capacity available is abundant. As we saw in Chapter 5, problems with similar structure to FASR are more
easily solved by CPLEX with excess capacity. Interestingly, when $\Theta$ was taken to be 1, the problems became much easier for CPLEX than branch-and-price. Of course, with $\Theta = 1$, the shared consumption component of the problem is drastically minimized. Therefore, the problem resembles a GAPFD with additional side constraints. From Chapter 4 we know that CPLEX outperforms branch-and-price for this class of problems if revenue functions are taken to be linear.

Another problem component that can be changed is the values $\phi_i$ ($i \in I$), which can be interpreted as the anticipated fraction of customer types that may be assigned to any individual facility. In our main results we set the values of $\phi_i$ ($i \in I$) at 0.5. We alternatively considered the impact of changing these values to 0.2 and 0.9. Tables 7-6 and 7-7 provide results for instances with much tighter facility capacities resulting from $\phi_i = 0.2$ ($i \in I$). When compared with results of Tables 7-1 and 7-2, the branch-and-price algorithm solves these more capacity-restricted instances in less time, on average. However, the time required by CPLEX increases notably. While the average time for branch-and-price to solve the 15 facility/45 customer instances decreases by more than 4 minutes, CPLEX solved none of these instances within one hour. The improved performance of the branch-and-price algorithm when $\phi_i = 0.2$ ($i \in I$) is expected since the number of feasible columns in MP-F is decreased. Alternatively, if we increase $\phi_i$ ($i \in I$) to 0.9, CPLEX consistently outperforms our branch-and-price algorithm. Table 7-8 shows that CPLEX requires an average of only 3 seconds to solve these ‘looser’ instances, while branch-and-price takes more than 2 minutes.

The last problem component that we consider is the customer type capacities, $g_q$ ($q \in Q$). In our base results, these capacities were generated with parameter $\tau_t = 1.2$. In Tables 7-9–7-12 we show the results of modifying this parameter. First, Tables 7-9 and 7-10 provide results for instances with customer type capacities generated with $\tau_t = 1.1$. Both branch-and-price and CPLEX solve these instances in less time than that required for instances generated with $\tau_t = 1.2$. However, branch-and-price still
solves the 15 facility/30 customer instances shown in Table 7-9 more than 5 times faster than CPLEX. This performance difference is even greater for the 15 facility/45 customer instances in Table 7-9. For these instances, branch-and-price requires less than a minute, on average, while CPLEX requires more than 13 minutes. Interestingly, if the customer type capacities are unbounded, as is the case for the instances in Tables 7-11 and 7-12, the branch-and-price algorithm again requires less time than that needed in the base results, while the time required by CPLEX increases. In fact, CPLEX fails to solve 7 out of 10 of the 45 customer instances within an hour. However, branch-and-price is able to solve all 10 of the 45 customer instances and more than half of the 75 customer instances within the allotted time. It should be noted that the performance of branch-and-price in Tables 7-11 and 7-12 is likely related to the change in the structure of MP-F when customer type capacities are unbounded. In this case, constraints 7–32 are effectively omitted. Importantly, this drastically reduces the number of columns required in MP-F. Without the complicating customer type capacity constraints, at most one column needs to be considered for a given subset of customer assignments. That is, in the absence of 7–32, the optimal customer demand fulfillment levels (i.e. \( v_{ij} (i \in I; j \in J) \)) for a fixed subset of assignments can easily be determined.

7.9 Conclusions and Future Research

In this chapter we considered a class of assignment problems that separate customers into disjoint sets. Customers of the same type (i.e. belonging to the same set) are assumed to share common production requirements. The proposed model considered non-linear resource consumption attributes among customers of the same type. In addition, further capacity restrictions limited the resource consumption of all customers associated with a particular type, independent of what facility is used to satisfy the demand. We proposed an exact branch-and-price algorithm to solve the resulting FASR problem based on a reformulation of our model as a set-partitioning representation with side constraints. This reformulation requires unique column representations to accurately model the
flexible demand component of the problem. To solve the resulting pricing problem, we studied a class of knapsack problems with an important relationship to the class of knapsack problems studied in Chapter 4. Our computational study suggests that the branch-and-price approach proposed in this work performs well in comparison to CPLEX on a large assortment of problem instances.

In future research, it may be advantageous to consider a slightly altered reformulation of FASR. Specifically, in this chapter the demand fulfillment levels associated with a column were determined by an extreme point solution to the following optimization problem

\[
\max \sum_{j \in J} r_{ij} v_{ij} + C_i^d
\]

subject to

\[
\sum_{j \in J} v_{ij} \leq \bar{b}_i^d
\]

\[
\ell_{ij} x_{ij} \leq v_{ij} \leq u_{ij} x_{ij}
\]

for a given subset of assignments. Recall that \( C_i^d = \sum_{j \in J} p_{ij} x_{ij}^d \) and \( \bar{b}_i^d = b_i - \sum_{q \in Q} f_q \max_{j \in J_q \{ x_{ij}^d \}} \). Theorem 10 established that using this column representation in conjunction with the appropriate constraints and decision variables yielded an equivalent representation of FASR. Altering the columns of the master problem so that demand fulfillment levels were determined by

\[
\max \sum_{j \in J} r_{ij} v_{ij} + C_i^d
\]

subject to

\[
\sum_{j \in J} v_{ij} \leq \bar{b}_i^d
\]

\[
\ell_{ij} x_{ij} \leq v_{ij} \leq u_{ij} x_{ij}
\]

for a given subset of assignments.
\[ \sum_{j \in J_q} v_{ij} \leq g_q \quad q \in Q \]  

with \( C_i^d \) and \( \bar{b}_i^d \) defined above, would yield an alternative pricing problem with an additional set constraints. The additional constraints (7–76) would potentially eliminate columns from MP-F that would be allowed in the procedure proposed in this chapter. An important question would be whether considering this more highly constrained pricing problem would yield a tighter linear relaxation of MP-F. If so, can the pricing problem be solved as efficiently? A computational study may reveal interesting tradeoffs between implementing the approach provided in this chapter versus an exact approach that utilizes this slightly modified column representation.

Figure 7-1. Illustration of \( r_q \) and \( \theta_q \)
Table 7-1. FASR: 15 facilities, 30 customers, 3 customer types, \( \tau_a = \tau_t = 1.2, \phi_i = .5 \) \((i \in \mathcal{I}), \Theta = 5\)

<table>
<thead>
<tr>
<th>Exp</th>
<th>Root Cols</th>
<th>Total Cols</th>
<th>BP Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>CPLEX Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>485</td>
<td>611</td>
<td>21</td>
<td>2.2</td>
<td>8.7</td>
<td>3.7</td>
</tr>
<tr>
<td>2</td>
<td>420</td>
<td>420</td>
<td>1</td>
<td>4.8</td>
<td>4.9</td>
<td>1.7</td>
</tr>
<tr>
<td>3</td>
<td>458</td>
<td>536</td>
<td>15</td>
<td>6.9</td>
<td>12.3</td>
<td>513.5</td>
</tr>
<tr>
<td>4</td>
<td>374</td>
<td>574</td>
<td>29</td>
<td>5.1</td>
<td>47.1</td>
<td>111.1</td>
</tr>
<tr>
<td>5</td>
<td>460</td>
<td>735</td>
<td>89</td>
<td>3.2</td>
<td>22.6</td>
<td>226.6</td>
</tr>
<tr>
<td>6</td>
<td>498</td>
<td>515</td>
<td>3</td>
<td>2.2</td>
<td>3.1</td>
<td>14.0</td>
</tr>
<tr>
<td>7</td>
<td>395</td>
<td>578</td>
<td>37</td>
<td>3.0</td>
<td>13.3</td>
<td>45.3</td>
</tr>
<tr>
<td>8</td>
<td>419</td>
<td>471</td>
<td>7</td>
<td>8.4</td>
<td>11.5</td>
<td>40.3</td>
</tr>
<tr>
<td>9</td>
<td>542</td>
<td>694</td>
<td>39</td>
<td>3.3</td>
<td>16.3</td>
<td>14.7</td>
</tr>
<tr>
<td>10</td>
<td>417</td>
<td>460</td>
<td>9</td>
<td>1.9</td>
<td>3.4</td>
<td>17.5</td>
</tr>
<tr>
<td>Avg</td>
<td>446.8</td>
<td>559.4</td>
<td>25</td>
<td>4.1</td>
<td>14.3</td>
<td>99.3</td>
</tr>
</tbody>
</table>

Table 7-2. FASR: 15 facilities, 45 customers, 3 customer types, \( \tau_a = \tau_t = 1.2, \phi_i = .5 \) \((i \in \mathcal{I}), \Theta = 5\)

<table>
<thead>
<tr>
<th>Exp</th>
<th>Root Cols</th>
<th>Total Cols</th>
<th>BP Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>CPLEX Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>797</td>
<td>2233</td>
<td>641</td>
<td>7.8</td>
<td>394.5</td>
<td>2059.0</td>
</tr>
<tr>
<td>2</td>
<td>1068</td>
<td>1205</td>
<td>13</td>
<td>12.2</td>
<td>25.6</td>
<td>133.2</td>
</tr>
<tr>
<td>3</td>
<td>820</td>
<td>2727</td>
<td>1005</td>
<td>4.8</td>
<td>785.5</td>
<td>3600.0(^{(17%)})</td>
</tr>
<tr>
<td>4</td>
<td>891</td>
<td>2637</td>
<td>1043</td>
<td>7.0</td>
<td>791.8</td>
<td>3600.0(^{(30%)})</td>
</tr>
<tr>
<td>5</td>
<td>1178</td>
<td>1747</td>
<td>93</td>
<td>8.4</td>
<td>107.8</td>
<td>535.5</td>
</tr>
<tr>
<td>6</td>
<td>771</td>
<td>1573</td>
<td>241</td>
<td>5.2</td>
<td>126.0</td>
<td>2038.0</td>
</tr>
<tr>
<td>7</td>
<td>926</td>
<td>1441</td>
<td>51</td>
<td>7.3</td>
<td>59.6</td>
<td>582.6</td>
</tr>
<tr>
<td>8</td>
<td>1082</td>
<td>1823</td>
<td>115</td>
<td>7.9</td>
<td>120.6</td>
<td>1480.8</td>
</tr>
<tr>
<td>9</td>
<td>967</td>
<td>2866</td>
<td>1483</td>
<td>6.7</td>
<td>1329.7</td>
<td>3600.0(^{(20%)})</td>
</tr>
<tr>
<td>10</td>
<td>863</td>
<td>1610</td>
<td>173</td>
<td>9.7</td>
<td>141.4</td>
<td>3600.0(^{(20%)})</td>
</tr>
<tr>
<td>Avg</td>
<td>936.3</td>
<td>1986.2</td>
<td>485.8</td>
<td>7.7</td>
<td>388.2</td>
<td>2124.2(^{(0.09%)})</td>
</tr>
</tbody>
</table>
Table 7-3. FASR: 30 facilities, 60 customers, 3 customer types, $\tau_a = \tau_t = 1.2$, $\phi_i = .5 \quad (i \in I), \Theta = 5$

<table>
<thead>
<tr>
<th>Exp</th>
<th>Root Cols</th>
<th>Total Cols</th>
<th>BP Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>CPLEX Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>879</td>
<td>1322</td>
<td>147</td>
<td>6.0</td>
<td>74.6</td>
<td>3600.0(18%)</td>
</tr>
<tr>
<td>2</td>
<td>812</td>
<td>1101</td>
<td>97</td>
<td>5.7</td>
<td>44.9</td>
<td>3600.0(12%)</td>
</tr>
<tr>
<td>3</td>
<td>817</td>
<td>916</td>
<td>25</td>
<td>4.3</td>
<td>13.3</td>
<td>3600.0(17%)</td>
</tr>
<tr>
<td>4</td>
<td>881</td>
<td>1093</td>
<td>131</td>
<td>10.1</td>
<td>60.3</td>
<td>3600.0(43%)</td>
</tr>
<tr>
<td>5</td>
<td>833</td>
<td>1022</td>
<td>39</td>
<td>5.5</td>
<td>18.3</td>
<td>3600.0(20%)</td>
</tr>
<tr>
<td>6</td>
<td>851</td>
<td>909</td>
<td>9</td>
<td>8.8</td>
<td>16.4</td>
<td>3600.0(32%)</td>
</tr>
<tr>
<td>7</td>
<td>833</td>
<td>889</td>
<td>9</td>
<td>7.3</td>
<td>11.0</td>
<td>3600.0(26%)</td>
</tr>
<tr>
<td>8</td>
<td>794</td>
<td>881</td>
<td>17</td>
<td>4.3</td>
<td>11.1</td>
<td>3600.0(17%)</td>
</tr>
<tr>
<td>9</td>
<td>798</td>
<td>849</td>
<td>11</td>
<td>6.0</td>
<td>12.3</td>
<td>3600.0(21%)</td>
</tr>
<tr>
<td>10</td>
<td>895</td>
<td>957</td>
<td>11</td>
<td>5.9</td>
<td>14.2</td>
<td>3600.0(32%)</td>
</tr>
<tr>
<td>Avg</td>
<td>839.3</td>
<td>993.9</td>
<td>49.6</td>
<td>6.4</td>
<td>27.7</td>
<td>3600.0(24%)</td>
</tr>
</tbody>
</table>

Table 7-4. FASR: 30 facilities, 90 customers, 3 customer types, $\tau_a = \tau_t = 1.2$, $\phi_i = .5 \quad (i \in I), \Theta = 5$

<table>
<thead>
<tr>
<th>Exp</th>
<th>Root Cols</th>
<th>Total Cols</th>
<th>BP Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>CPLEX Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2939</td>
<td>3520</td>
<td>101</td>
<td>23.1</td>
<td>305.0</td>
<td>3600.0(41%)</td>
</tr>
<tr>
<td>2</td>
<td>2808</td>
<td>3099</td>
<td>33</td>
<td>23.2</td>
<td>111.2</td>
<td>3600.0(46%)</td>
</tr>
<tr>
<td>3</td>
<td>2614</td>
<td>2799</td>
<td>11</td>
<td>18.4</td>
<td>53.2</td>
<td>3600.0(17%)</td>
</tr>
<tr>
<td>4</td>
<td>2476</td>
<td>2681</td>
<td>17</td>
<td>22.4</td>
<td>70.6</td>
<td>3600.0(25%)</td>
</tr>
<tr>
<td>5</td>
<td>2682</td>
<td>3660</td>
<td>203</td>
<td>24.9</td>
<td>557.9</td>
<td>3600.0(35%)</td>
</tr>
<tr>
<td>6</td>
<td>2005</td>
<td>3100</td>
<td>17</td>
<td>21.7</td>
<td>70.0</td>
<td>3600.0(27%)</td>
</tr>
<tr>
<td>7</td>
<td>3337</td>
<td>3696</td>
<td>51</td>
<td>28.8</td>
<td>152.0</td>
<td>3600.0(21%)</td>
</tr>
<tr>
<td>8</td>
<td>2602</td>
<td>4363</td>
<td>473</td>
<td>23.7</td>
<td>1215.0</td>
<td>3600.0(39%)</td>
</tr>
<tr>
<td>9</td>
<td>2086</td>
<td>3193</td>
<td>97</td>
<td>16.2</td>
<td>229.4</td>
<td>3600.0(30%)</td>
</tr>
<tr>
<td>10</td>
<td>3177</td>
<td>5061</td>
<td>1083</td>
<td>19.3</td>
<td>3600.0</td>
<td>3600.0(41%)</td>
</tr>
<tr>
<td>Avg</td>
<td>2762.6</td>
<td>3517.2</td>
<td>208.6</td>
<td>22.2</td>
<td>636.6</td>
<td>3600.0(32%)</td>
</tr>
</tbody>
</table>
Table 7-5. FASR: 15 facilities, 45 customers, 3 customer types, $\tau_a = \tau_t = 1.2$, $\phi_i = .5$
$(i \in I), \Theta = 25$

<table>
<thead>
<tr>
<th>Exp</th>
<th>Root Cols</th>
<th>Total Cols</th>
<th>BP Cols</th>
<th>BP Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>CPLEX Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>557</td>
<td>581</td>
<td>9</td>
<td></td>
<td>2.7</td>
<td>4.0</td>
<td>31.9</td>
</tr>
<tr>
<td>2</td>
<td>648</td>
<td>682</td>
<td>9</td>
<td></td>
<td>3.1</td>
<td>4.8</td>
<td>61.0</td>
</tr>
<tr>
<td>3</td>
<td>569</td>
<td>569</td>
<td>1</td>
<td></td>
<td>2.3</td>
<td>2.4</td>
<td>23.8</td>
</tr>
<tr>
<td>4</td>
<td>579</td>
<td>893</td>
<td>67</td>
<td></td>
<td>2.0</td>
<td>17.4</td>
<td>266.0</td>
</tr>
<tr>
<td>5</td>
<td>604</td>
<td>874</td>
<td>25</td>
<td></td>
<td>3.5</td>
<td>15.2</td>
<td>91.1</td>
</tr>
<tr>
<td>6</td>
<td>621</td>
<td>853</td>
<td>59</td>
<td></td>
<td>2.3</td>
<td>12.7</td>
<td>181.3</td>
</tr>
<tr>
<td>7</td>
<td>681</td>
<td>681</td>
<td>1</td>
<td></td>
<td>2.4</td>
<td>2.4</td>
<td>15.2</td>
</tr>
<tr>
<td>8</td>
<td>639</td>
<td>660</td>
<td>3</td>
<td></td>
<td>2.5</td>
<td>3.1</td>
<td>15.2</td>
</tr>
<tr>
<td>9</td>
<td>594</td>
<td>991</td>
<td>51</td>
<td></td>
<td>2.6</td>
<td>21.0</td>
<td>301.8</td>
</tr>
<tr>
<td>10</td>
<td>603</td>
<td>715</td>
<td>11</td>
<td></td>
<td>1.7</td>
<td>4.5</td>
<td>70.1</td>
</tr>
<tr>
<td>Avg</td>
<td>609.5</td>
<td>749.9</td>
<td>23.6</td>
<td></td>
<td>2.5</td>
<td>8.7</td>
<td>105.7</td>
</tr>
</tbody>
</table>

Table 7-6. FASR: 15 facilities, 30 customers, 3 customer types, $\tau_a = \tau_t = 1.2$, $\phi_i = .2$
$(i \in I), \Theta = 5$

<table>
<thead>
<tr>
<th>Exp</th>
<th>Root Cols</th>
<th>Total Cols</th>
<th>BP Cols</th>
<th>BP Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>CPLEX Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>317</td>
<td>317</td>
<td>1</td>
<td></td>
<td>6.8</td>
<td>6.8</td>
<td>87.2</td>
</tr>
<tr>
<td>2</td>
<td>309</td>
<td>351</td>
<td>9</td>
<td></td>
<td>9.1</td>
<td>14.8</td>
<td>88.2</td>
</tr>
<tr>
<td>3</td>
<td>286</td>
<td>354</td>
<td>21</td>
<td></td>
<td>11.2</td>
<td>24.0</td>
<td>645.2</td>
</tr>
<tr>
<td>4</td>
<td>310</td>
<td>310</td>
<td>1</td>
<td></td>
<td>8.8</td>
<td>8.8</td>
<td>35.9</td>
</tr>
<tr>
<td>5</td>
<td>291</td>
<td>291</td>
<td>1</td>
<td></td>
<td>10.5</td>
<td>10.5</td>
<td>89.5</td>
</tr>
<tr>
<td>6</td>
<td>319</td>
<td>357</td>
<td>7</td>
<td></td>
<td>9.8</td>
<td>16.1</td>
<td>739.2</td>
</tr>
<tr>
<td>7</td>
<td>322</td>
<td>339</td>
<td>9</td>
<td></td>
<td>13.5</td>
<td>17.9</td>
<td>404.2</td>
</tr>
<tr>
<td>8</td>
<td>303</td>
<td>303</td>
<td>1</td>
<td></td>
<td>11.9</td>
<td>11.9</td>
<td>93.3</td>
</tr>
<tr>
<td>9</td>
<td>308</td>
<td>308</td>
<td>1</td>
<td></td>
<td>11.7</td>
<td>11.7</td>
<td>161.6</td>
</tr>
<tr>
<td>10</td>
<td>306</td>
<td>306</td>
<td>1</td>
<td></td>
<td>10.7</td>
<td>10.7</td>
<td>68.4</td>
</tr>
<tr>
<td>Avg</td>
<td>307.1</td>
<td>323.6</td>
<td>5.2</td>
<td></td>
<td>10.4</td>
<td>13.3</td>
<td>241.4</td>
</tr>
</tbody>
</table>
Table 7-7. FASR: 15 facilities, 45 customers, 3 customer types, \( \tau_a = \tau_t = 1.2, \phi_i = .2 \) 
\((i \in \mathcal{I}), \Theta = 5\)

<table>
<thead>
<tr>
<th>Exp</th>
<th>Root Cols</th>
<th>Total Cols</th>
<th>BP Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>CPLEX Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>594</td>
<td>656</td>
<td>3</td>
<td>30.1</td>
<td>48.1</td>
<td>3600.0 (5%)</td>
</tr>
<tr>
<td>2</td>
<td>624</td>
<td>964</td>
<td>105</td>
<td>26.9</td>
<td>138.6</td>
<td>3600.0 (5%)</td>
</tr>
<tr>
<td>3</td>
<td>605</td>
<td>681</td>
<td>15</td>
<td>48.3</td>
<td>73.7</td>
<td>3600.0 (4%)</td>
</tr>
<tr>
<td>4</td>
<td>586</td>
<td>723</td>
<td>23</td>
<td>74.7</td>
<td>215.2</td>
<td>3600.0 (7%)</td>
</tr>
<tr>
<td>5</td>
<td>656</td>
<td>872</td>
<td>33</td>
<td>33.9</td>
<td>119.1</td>
<td>3600.0 (5%)</td>
</tr>
<tr>
<td>6</td>
<td>586</td>
<td>692</td>
<td>25</td>
<td>31.0</td>
<td>80.4</td>
<td>3600.0 (5%)</td>
</tr>
<tr>
<td>7</td>
<td>616</td>
<td>616</td>
<td>1</td>
<td>41.2</td>
<td>41.3</td>
<td>3600.0 (5%)</td>
</tr>
<tr>
<td>8</td>
<td>592</td>
<td>653</td>
<td>9</td>
<td>52.3</td>
<td>88.7</td>
<td>3600.0 (7%)</td>
</tr>
<tr>
<td>9</td>
<td>653</td>
<td>833</td>
<td>37</td>
<td>212.6</td>
<td>292.2</td>
<td>3600.0 (7%)</td>
</tr>
<tr>
<td>10</td>
<td>595</td>
<td>860</td>
<td>49</td>
<td>36.0</td>
<td>148.3</td>
<td>3600.0 (5%)</td>
</tr>
<tr>
<td>Avg</td>
<td>610.7</td>
<td>755</td>
<td>30</td>
<td>58.7</td>
<td>124.5</td>
<td>3600.0 (6%)</td>
</tr>
</tbody>
</table>

Table 7-8. FASR: 15 facilities, 75 customers, 3 customer types, \( \tau_a = \tau_t = 1.2, \phi_i = .9 \) 
\((i \in \mathcal{I}), \Theta = 5\)

<table>
<thead>
<tr>
<th>Exp</th>
<th>Root Cols</th>
<th>Total Cols</th>
<th>BP Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>CPLEX Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3706</td>
<td>4354</td>
<td>55</td>
<td>25.2</td>
<td>201.8</td>
<td>14.2</td>
</tr>
<tr>
<td>2</td>
<td>3608</td>
<td>3892</td>
<td>13</td>
<td>20.5</td>
<td>73.5</td>
<td>1.3</td>
</tr>
<tr>
<td>3</td>
<td>2219</td>
<td>2527</td>
<td>13</td>
<td>16.3</td>
<td>51.5</td>
<td>1.4</td>
</tr>
<tr>
<td>4</td>
<td>3414</td>
<td>4078</td>
<td>55</td>
<td>22.5</td>
<td>171.0</td>
<td>1.4</td>
</tr>
<tr>
<td>5</td>
<td>3276</td>
<td>3912</td>
<td>43</td>
<td>23.1</td>
<td>146.7</td>
<td>1.9</td>
</tr>
<tr>
<td>6</td>
<td>3398</td>
<td>4269</td>
<td>37</td>
<td>25.3</td>
<td>169.2</td>
<td>3.6</td>
</tr>
<tr>
<td>7</td>
<td>3900</td>
<td>4445</td>
<td>27</td>
<td>25.0</td>
<td>150.7</td>
<td>2.3</td>
</tr>
<tr>
<td>8</td>
<td>3447</td>
<td>4057</td>
<td>43</td>
<td>21.9</td>
<td>154.2</td>
<td>1.4</td>
</tr>
<tr>
<td>9</td>
<td>2325</td>
<td>3283</td>
<td>55</td>
<td>.14.5</td>
<td>108.7</td>
<td>1.7</td>
</tr>
<tr>
<td>10</td>
<td>2466</td>
<td>3078</td>
<td>29</td>
<td>16.9</td>
<td>75.3</td>
<td>1.0</td>
</tr>
<tr>
<td>Avg</td>
<td>3175.9</td>
<td>3789.5</td>
<td>37</td>
<td>21.1</td>
<td>130.3</td>
<td>3.0</td>
</tr>
</tbody>
</table>
### Table 7-9. FASR: 15 facilities, 30 customers, 3 customer types, $\tau_a = 1.2$, $\tau_t = 1.1$, $\phi_i = .5$ ($i \in I$), $\Theta = 5$

<table>
<thead>
<tr>
<th>Exp</th>
<th>Root Cols</th>
<th>Total Cols</th>
<th>BP Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>CPLEX Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>350</td>
<td>448</td>
<td>17</td>
<td>2.4</td>
<td>8.2</td>
<td>6.6</td>
</tr>
<tr>
<td>2</td>
<td>406</td>
<td>406</td>
<td>1</td>
<td>1.3</td>
<td>1.3</td>
<td>2.3</td>
</tr>
<tr>
<td>3</td>
<td>440</td>
<td>504</td>
<td>13</td>
<td>3.7</td>
<td>10.9</td>
<td>227.0</td>
</tr>
<tr>
<td>4</td>
<td>403</td>
<td>429</td>
<td>3</td>
<td>4.1</td>
<td>5.4</td>
<td>1.3</td>
</tr>
<tr>
<td>5</td>
<td>463</td>
<td>482</td>
<td>3</td>
<td>3.5</td>
<td>4.6</td>
<td>27.1</td>
</tr>
<tr>
<td>6</td>
<td>396</td>
<td>396</td>
<td>1</td>
<td>3.1</td>
<td>3.1</td>
<td>8.9</td>
</tr>
<tr>
<td>7</td>
<td>338</td>
<td>363</td>
<td>3</td>
<td>1.7</td>
<td>3.0</td>
<td>34.9</td>
</tr>
<tr>
<td>8</td>
<td>417</td>
<td>437</td>
<td>5</td>
<td>4.8</td>
<td>6.1</td>
<td>5.6</td>
</tr>
<tr>
<td>9</td>
<td>351</td>
<td>582</td>
<td>109</td>
<td>2.1</td>
<td>19.6</td>
<td>36.1</td>
</tr>
<tr>
<td>10</td>
<td>422</td>
<td>422</td>
<td>1</td>
<td>2.1</td>
<td>2.1</td>
<td>11.7</td>
</tr>
<tr>
<td>Avg</td>
<td>398.6</td>
<td>446.9</td>
<td>15.6</td>
<td>2.9</td>
<td>6.5</td>
<td>36.1</td>
</tr>
</tbody>
</table>

### Table 7-10. FASR: 15 facilities, 45 customers, 3 customer types, $\tau_a = 1.2$, $\tau_t = 1.1$, $\phi_i = .5$ ($i \in I$), $\Theta = 5$

<table>
<thead>
<tr>
<th>Exp</th>
<th>Root Cols</th>
<th>Total Cols</th>
<th>BP Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>CPLEX Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>794</td>
<td>846</td>
<td>7</td>
<td>7.5</td>
<td>11.2</td>
<td>231.2</td>
</tr>
<tr>
<td>2</td>
<td>737</td>
<td>888</td>
<td>19</td>
<td>6.3</td>
<td>17.6</td>
<td>117.4</td>
</tr>
<tr>
<td>3</td>
<td>1052</td>
<td>1469</td>
<td>59</td>
<td>10.6</td>
<td>66.8</td>
<td>618.0</td>
</tr>
<tr>
<td>4</td>
<td>931</td>
<td>1399</td>
<td>53</td>
<td>9.7</td>
<td>52.0</td>
<td>1283.9</td>
</tr>
<tr>
<td>5</td>
<td>1144</td>
<td>1756</td>
<td>131</td>
<td>11.4</td>
<td>124.6</td>
<td>846.4</td>
</tr>
<tr>
<td>6</td>
<td>1021</td>
<td>1356</td>
<td>31</td>
<td>5.0</td>
<td>30.0</td>
<td>24.9</td>
</tr>
<tr>
<td>7</td>
<td>980</td>
<td>1094</td>
<td>17</td>
<td>6.5</td>
<td>20.2</td>
<td>930.4</td>
</tr>
<tr>
<td>8</td>
<td>809</td>
<td>1257</td>
<td>77</td>
<td>8.1</td>
<td>75.4</td>
<td>3600.0(18%)</td>
</tr>
<tr>
<td>9</td>
<td>891</td>
<td>1489</td>
<td>127</td>
<td>9.0</td>
<td>77.9</td>
<td>275.8</td>
</tr>
<tr>
<td>10</td>
<td>772</td>
<td>1409</td>
<td>79</td>
<td>4.2</td>
<td>58.7</td>
<td>274.2</td>
</tr>
<tr>
<td>Avg</td>
<td>913.1</td>
<td>1296.3</td>
<td>60</td>
<td>7.8</td>
<td>53.4</td>
<td>820.2(18%)</td>
</tr>
</tbody>
</table>
Table 7-11. FASR: 15 facilities, 30 customers, 3 customer types, $\tau_a = 1.2$, $\phi_i = .5$ ($i \in I$), $\Theta = 5$, $g_q = \infty$ ($q \in Q$)

<table>
<thead>
<tr>
<th>Exp</th>
<th>Root Cols</th>
<th>Total Cols</th>
<th>BP Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>CPLEX Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>356</td>
<td>356</td>
<td>1</td>
<td>10.9</td>
<td>10.9</td>
<td>618.3</td>
</tr>
<tr>
<td>2</td>
<td>364</td>
<td>416</td>
<td>5</td>
<td>7.1</td>
<td>18.6</td>
<td>23.6</td>
</tr>
<tr>
<td>3</td>
<td>387</td>
<td>387</td>
<td>1</td>
<td>2.9</td>
<td>2.9</td>
<td>64.7</td>
</tr>
<tr>
<td>4</td>
<td>350</td>
<td>369</td>
<td>5</td>
<td>4.3</td>
<td>8.3</td>
<td>719.7</td>
</tr>
<tr>
<td>5</td>
<td>376</td>
<td>428</td>
<td>21</td>
<td>5.2</td>
<td>13.3</td>
<td>430.3</td>
</tr>
<tr>
<td>6</td>
<td>351</td>
<td>366</td>
<td>3</td>
<td>4.1</td>
<td>10.3</td>
<td>206.0</td>
</tr>
<tr>
<td>7</td>
<td>382</td>
<td>382</td>
<td>1</td>
<td>8.3</td>
<td>8.4</td>
<td>67.1</td>
</tr>
<tr>
<td>8</td>
<td>352</td>
<td>374</td>
<td>9</td>
<td>5.2</td>
<td>9.5</td>
<td>750.6</td>
</tr>
<tr>
<td>9</td>
<td>353</td>
<td>431</td>
<td>31</td>
<td>3.6</td>
<td>16.9</td>
<td>906.4</td>
</tr>
<tr>
<td>10</td>
<td>379</td>
<td>415</td>
<td>11</td>
<td>2.1</td>
<td>7.1</td>
<td>1011.7</td>
</tr>
<tr>
<td>Avg</td>
<td>365</td>
<td>392.4</td>
<td>8.8</td>
<td>5.4</td>
<td>10.6</td>
<td>479.8</td>
</tr>
</tbody>
</table>

Table 7-12. FASR: 15 facilities, 45 customers, 3 customer types, $\tau_a = 1.2$, $\phi_i = .5$ ($i \in I$), $\Theta = 5$, $g_q = \infty$ ($q \in Q$)

<table>
<thead>
<tr>
<th>Exp</th>
<th>Root Cols</th>
<th>Total Cols</th>
<th>BP Nodes</th>
<th>Root time (sec)</th>
<th>BP total time (sec)</th>
<th>CPLEX Time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>596</td>
<td>724</td>
<td>45</td>
<td>41.2</td>
<td>116.0</td>
<td>3600.0(1.5%)</td>
</tr>
<tr>
<td>2</td>
<td>707</td>
<td>741</td>
<td>5</td>
<td>33.4</td>
<td>44.7</td>
<td>1278.7</td>
</tr>
<tr>
<td>3</td>
<td>654</td>
<td>814</td>
<td>27</td>
<td>42.8</td>
<td>111.7</td>
<td>3600.0(3.5%)</td>
</tr>
<tr>
<td>4</td>
<td>655</td>
<td>827</td>
<td>31</td>
<td>37.9</td>
<td>87.2</td>
<td>3600.0(85%)</td>
</tr>
<tr>
<td>5</td>
<td>659</td>
<td>788</td>
<td>33</td>
<td>48.2</td>
<td>93.8</td>
<td>3600.0(1.4%)</td>
</tr>
<tr>
<td>6</td>
<td>572</td>
<td>617</td>
<td>9</td>
<td>33.6</td>
<td>45.8</td>
<td>3600.0(1.0%)</td>
</tr>
<tr>
<td>7</td>
<td>776</td>
<td>911</td>
<td>25</td>
<td>32.7</td>
<td>69.6</td>
<td>3473.6</td>
</tr>
<tr>
<td>8</td>
<td>723</td>
<td>723</td>
<td>1</td>
<td>26.6</td>
<td>26.7</td>
<td>2461.4</td>
</tr>
<tr>
<td>9</td>
<td>625</td>
<td>798</td>
<td>31</td>
<td>58.8</td>
<td>118.2</td>
<td>3600.0(1.4%)</td>
</tr>
<tr>
<td>10</td>
<td>624</td>
<td>768</td>
<td>37</td>
<td>34.5</td>
<td>83.7</td>
<td>3600.0(79%)</td>
</tr>
<tr>
<td>Avg</td>
<td>659.1</td>
<td>771.1</td>
<td>24.4</td>
<td>39.0</td>
<td>79.8</td>
<td>3241.4(74%)</td>
</tr>
</tbody>
</table>
CHAPTER 8
CONCLUSION

In this dissertation we explored numerous variants of resource constrained assignment problems that account for real-world operations decisions. In each of the problems considered, the optimization models seek to exploit lesser studied relationships between customers and the manufacturer to increase a firm’s profit. In Chapter 3, we introduced a generalization of the capacitated facility location with single-sourcing constraints. A notable feature of this model was the allowance of flexible customer demand, which has received little attention in the literature. We considered variants of the problem with and without the requirement that resources must be procured at a fixed cost to the decision maker. In Chapter 4, we provided an exact branch-and-price algorithm based on a reformulation of the model that solved both problems. Our approach required the study of an interesting class of knapsack problems with flexible demand. We showed important structural results of a relaxation of this class of knapsack problems that led to an efficient solution approach for our pricing problem with generalized revenue functions. We offered even more efficient algorithms for solving instances with specially structured revenue functions that correspond to common pricing structures. The computational study of our branch-and-price algorithm demonstrated the value of our approach. A detailed discussion of the implementation choices that resulted in reduced solution times was provided. Elements of this discussion were relevant to all fixed-charge problems solved using column generation.

In Chapters 5 and 6, we developed heuristics to solve large-scale instances of the problem variants with (CFLFD) and without (GAPFD) resource procurement decisions. In Chapter 5, we proposed a class of greedy heuristics for GAPFD with linear revenue functions that was motivated by properties of an optimal solution to the linear relaxation of our model. We presented a novel perturbation scheme that guaranteed our class of heuristics was asymptotically optimality under a very general stochastic model. Our
computational study demonstrated that our heuristic performed particularly well for instances with a large ratio of customers-to-facilities. Applying the concept of instance perturbation in developing heuristics for other capacitated assignment problems would be interesting. Comparing the success of such heuristics on large-scale problems versus established procedures would be especially important for validating the contribution of perturbation in cases where it yields strong performance guarantees.

In Chapter 6, we developed a large-scale search heuristic for CFLFD with linear revenue functions. Our approach utilized the high-quality efficient heuristic proposed for GAPFD within a facility neighborhood search to address the combined assignment and fixed-charge structure of our underlying optimization problem. We also considered the advantages of developing a hybrid approach that utilized a so-called very large-scale neighborhood search (VLSN) method. Our computational results indicated that our heuristic framework was an effective approach for solving CFLFD. It would be interesting to apply this heuristic framework to other fixed-charge assignment problems. In addition, since the heuristic calls for assignments to be made using a secondary procedure (in this case, the GAPFD heuristic) it would be interesting to consider additional heuristics and exact approaches in this phase.

Lastly, Chapter 7 introduced an additional class of assignment problems with non-linear capacity consumption among customers and capacity constraints that spanned all available resources. This model was applicable to production scenarios that consider products with similar production requirements. The additional capacity constraints accounted for real-world limitations on hazard emissions, logistics resources or warehouse space. The branch-and-price algorithm developed for this class of problems required an interesting reformulation of our problem that included columns with a unique representation when compared to those typically seen in assignment problems. The subproblem to be solved resulted in a study of another class of knapsack problems with an important relationship to the knapsack problems studied in the case of CFLFD and
A computational study demonstrated the advantages of our approach over a well-known commercial solver. Generalizing FASR to allow for a shared profit associated with each customer type is a natural extension. Furthermore, while the branch-and-price approach was shown to be successful on instances with linear revenue functions, the ability to solve instances with general revenue functions may result in a greater impact. For that reason, the class of knapsack problems denoted by SKFP should be studied with additional non-linear considerations. Of particular interest would be how MP-F might be modified to maintain its equivalence with a non-linear representation of FASR.

Lastly, while the notion of customer sets (i.e. types) was studied in depth in this dissertation, another interesting generalization of CFLFD would be to group facilities into disjoint sets. Each of these sets may be limited by its own capacity restriction, in addition to the individual capacity of the facilities belonging to the sets. Along with procuring each facility, a procurement decision associated with the set must be considered as well. The additional consideration of facility sets is applicable to many production and personnel planning scenarios. For example, assume that facilities represent individual machines at various manufacturing locations. Each machine has a maximum amount of time that it can be run each day. However, the products produced by the machines at each location must be stored at an on-site warehouse before being transported to the centralized distributor. The space available in each warehouse is limited such that if each machine is run for its maximum time, the amount of products produced will exceed the space need to store them. Therefore, an additional constraint on the total number of hours each set of machines is producing is necessary. This very general scenario is applicable across a variety of industries. Therefore, this potential class of problems is rich in applications. Interestingly, it is possible that the neighborhood search heuristic used to solve CFLFD may be extended to this multi-level fixed-charge problem.
**APPENDIX A**

**GAPFD ASYMPTOTIC PROPERTY**

**Lemma 5** The optimal values of \((LP')\) and \((LP'(|J|))\) are close in the sense that, with probability one,

\[
\lim_{|J| \to \infty} \frac{1}{|J|} Z_{|J|}^{LP'}(\delta_{|J|}) = \lim_{|J| \to \infty} \frac{1}{|J|} Z_{|J|}^{LP'}.
\]

**Proof.** The normalized optimal value of \((LP'(|J|))\) can be expressed as

\[
\frac{1}{|J|} Z_{|J|}^{LP'}(\delta_{|J|}) = \min_{\lambda \geq 0} \Phi_{|J|}(\lambda; \delta_{|J|})
\]

where

\[
\Phi_{|J|}(\lambda; \delta_{|J|}) = \frac{1}{|J|} \sum_{j \in J} \max_{i \in I} f_\lambda(i, j) + \sum_{i \in I} \lambda_i (\beta_i - \delta_{|J|})
\]

\[
= \Phi_{|J|}(\lambda; 0) - \left( \sum_{i \in I} \lambda_i \right) \delta_{|J|}. \tag{A-1}
\]

We will show that we may restrict ourselves to vectors \(\lambda\) in a compact set. First, note that

\[
\min_{\lambda \geq 0} \Phi_{|J|}(\lambda; \delta) \leq \Phi_{|J|}(0; \delta) \leq \bar{R}(\bar{L} + \bar{D}) + \bar{P}.
\]

Furthermore, for any \(\lambda\) we have

\[
\Phi_{|J|}(\lambda; \delta)
\]

\[
= \frac{1}{|J|} \sum_{j \in J} \max_{i \in I} (r_{ij} - \lambda_i) e_{ij} + (r_{ij} - \lambda_i) e_{ij} + p_{ij} - \lambda_i a_{ij}) + \sum_{i \in I} \lambda_i \beta_i - \delta \cdot \sum_{i \in I} \lambda_i
\]

\[
\geq \frac{1}{|J|} \sum_{j \in J} \max_{i \in I} (p_{ij} - \lambda_i a_{ij} + (r_{ij} - \lambda_i) e_{ij}) + \sum_{i \in I} \lambda_i \beta_i - \delta \cdot \sum_{i \in I} \lambda_i
\]

\[
\geq RL + P + \left( \sum_{i \in I} \lambda_i \beta_i - \frac{1}{|J|} \sum_{j \in J} \min_{i \in I} \lambda_i (a_{ij} + e_{ij}) \right) - \delta \cdot \sum_{i \in I} \lambda_i
\]

\[
\geq RL + P + \min_{\lambda \geq 0, \lambda e = 1} \left( \sum_{i \in I} \lambda_i \beta_i - \frac{1}{|J|} \sum_{j \in J} \min_{i \in I} \lambda'_i (a_{ij} + e_{ij}) \right) \cdot \sum_{i \in I} \lambda_i - \delta \cdot \sum_{i \in I} \lambda_i
\]

\[
\rightarrow RL + P + (\Delta - \delta) \cdot \sum_{i \in I} \lambda_i.
\]
with probability one as $|\mathcal{J}| \to \infty$, by Romeijn and Piersma [78, Theorem 3.1]. Since the value $Z_{|\mathcal{J}|}^{LP}(\delta_{|\mathcal{J}|})$ is nonnegative, the function $\Phi_{|\mathcal{J}|}(\lambda; \delta)$ thus attains its minimum on the compact set $\Lambda = \{ \lambda \geq 0 : \sum_{i \in \mathcal{I}} \lambda_i \leq \Gamma \}$ (where $\Gamma = (R(L + D) + P - RL - P) / (\Delta - \delta)$) with probability one as $|\mathcal{J}| \to \infty$.

Now note that $\Phi_{|\mathcal{J}|}(\lambda; \delta_{|\mathcal{J}|}) \geq \Phi_{|\mathcal{J}|}(\lambda; \delta)$. By the convexity of $\Phi_{|\mathcal{J}|}(\lambda; 0)$ in $\lambda$ and equation (A–1) it then follows that the function $\Phi_{|\mathcal{J}|}(\lambda; \delta_{|\mathcal{J}|})$ also attains its minimum on $\Lambda$ with probability one as $|\mathcal{J}| \to \infty$. This means that

$$\Phi_{|\mathcal{J}|}(\lambda; \delta_{|\mathcal{J}|}) \geq \Phi_{|\mathcal{J}|}(\lambda; 0) - \Gamma \delta_{|\mathcal{J}|}$$

with probability one as $|\mathcal{J}| \to \infty$ so that

$$\frac{1}{|\mathcal{J}|} Z_{|\mathcal{J}|}^{LP} \geq \min_{\lambda \geq 0} \Phi_{|\mathcal{J}|}(\lambda; \delta_{|\mathcal{J}|})$$

$$\geq \min_{\lambda \geq 0} \Phi_{|\mathcal{J}|}(\lambda; 0) - \Gamma \delta_{|\mathcal{J}|}$$

$$= \frac{1}{|\mathcal{J}|} Z_{|\mathcal{J}|}^{LP} - \Gamma \delta_{|\mathcal{J}|}$$

with probability one as $|\mathcal{J}| \to \infty$.

The desired result now follows by using (5–20). \qed
APPENDIX B
CFLFD PRICING PROBLEM PROPERTY

**Theorem 1** The optimization problems (RKP′) and (KPEI′-R) are equivalent when the revenue function \( r_j \) are linear for all \( (j \in \tilde{J}) \).

**Proof.** We can rewrite the interval constraints (4–21) as follows:

\[
x_j \in \left[ \frac{w_j}{u'_j - \ell'_j}, \frac{w_j}{u'_j} \right] \quad j \in \tilde{J}.
\] (4–21′)

Clearly, we would like to choose the value of \( x_j \) as small as possible (i.e., \( x_j = \frac{w_j}{u'_j} \)) if \( \bar{p}_j \leq 0 \) and as large as possible (i.e., \( x_j = \min\{1, \frac{w_j}{\ell'_j}\} \)) if \( \bar{p}_j > 0 \). Therefore, if we define \( J^- = \{ j \in \tilde{J} : \bar{p}_j \leq 0 \} \) and \( J^+ = \{ j \in \tilde{J} : \bar{p}_j > 0 \} \), we can formulate the LP-relaxation of the (KPEI) as

\[
\max \quad \sum_{j \in J^-} \left( r_j + \bar{p}_j \right) w_j + \sum_{j \in J^+} \min \left\{ r_j w_j + \bar{p}_j, \left( r_j + \bar{p}_j \right) w_j \right\}
\]

subject to \( (\text{KPEI}'-\text{R}) \)

\[
\sum_{j \in J^-} w_j + \sum_{j \in J^+} w_j^1 + \sum_{j \in J^+} w_j^2 \leq b_i \quad j \in \tilde{J}.
\]

The revenue functions of items \( j \in J^+ \) are no longer linear, but instead a concave function of the job size \( w_j \). Each job \( j \in J^+ \) can be split into two parts. The first part has job size \( w_{j1} \in [0, \ell'_{ij}] \) and a revenue function given by \( \left( r_j + \frac{\bar{p}_j}{u'_j} \right) w_{j1} \). The second part has job size \( w_{j2} \in [0, u'_j - \ell'_j] \) and a revenue function given by \( r_j w_{j2} \). Therefore, (KPEI′-R) can alternatively be written as

\[
\max \quad \sum_{j \in J^-} \left( r_j + \bar{p}_j \right) w_j + \sum_{j \in J^+} \left( r_j + \bar{p}_j \right) w_{j1} + \sum_{j \in J^+} r_j w_{j2}
\]

subject to \( (\text{KPEI}'-\text{R}') \)

\[
\sum_{j \in J^-} w_j + \sum_{j \in J^+} w_{j1} + \sum_{j \in J^+} w_{j2} \leq b_i
\]

189
\[
    w_j \in [0, u'_j] \quad j \in J^-
\]
\[
    w_{j1} \in [0, \ell'_j] \quad j \in J^+
\]
\[
    w_{j2} \in [0, u'_j - \ell'_j] \quad j \in J^+.
\]

Note that
\[
    \left( r_{ij} + \bar{p}_j \right) = \left( \frac{r_{j}u'_j + \bar{p}_j}{u'_j} \right) = \frac{\phi_j(u'_j)}{u'_j} \quad (B-1)
\]
and
\[
    \left( r_j + \bar{p}_j \right) = \left( \frac{r_{j} \ell'_j + \bar{p}_j}{\ell'_j} \right) = \frac{\phi_{ij}(\ell'_{ij})}{\ell'_{ij}} = \alpha_{ij} \quad (B-2)
\]
and
\[
    r_j = \frac{\bar{p}_j + r_{j}u'_j - \bar{p}_j - r_{j} \ell'_j}{u'_j - \ell'_j} = \frac{\phi_j(u'_j) - \phi_j(\ell'_j)}{u'_j - \ell'_j} = \gamma_j. \quad (B-3)
\]

Therefore, the optimization problem (KPEI'-R') is precisely (RKP') presented in Section 4.2.2, which yields the desired result. \[\square\]
REFERENCES


the multilevel generalized assignment problem. *European Journal of Operational
Research* 82 176–189.

[58] Laporte, G. 1992. The vehicle routing problem: an overview of exact and


[60] Li, C. 1995. Scheduling to minimize the total resource consumption with a constraint
on the sum of completion times. *European Journal of Operational Research* 80
381–388.


and a branch and bound algorithm. *European Journal of Operational Research* 1
169–175.


[65] Martello, S., P. Toth. 1990. Dynamic programming and strong bounds for the 0-1


BIOGRAPHICAL SKETCH

Chase Rainwater was born in Mountain Home, AR. He graduated from Mountain Home High School in 1999. He attended the University of Arkansas, where he earned a Bachelor of Science in Industrial Engineering in May 2004. After his undergraduate studies, he married Candace Zieleniuk-Rainwater and enrolled in the graduate program at the University of Florida. He began his doctoral studies in August 2005 in the Industrial and Systems Engineering Department. He earned his Doctor of Philosophy in industrial and systems engineering in August 2009. Following graduation, he will join the faculty of the Department of Industrial Engineering at the University of Arkansas.