Solving Single and Multiple Plant Sourcing Problems with a Multidimensional Knapsack Model

Natalie S. Cherbaka

Dissertation submitted to the Faculty of
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Industrial and Systems Engineering

Russell D. Meller, Chair
Ebru K. Bish
F. Frank Chen
Barbara J. Hoopes
Hanif D. Sherali

November 18, 2004
Blacksburg, VA

Keywords: Multidimensional Knapsack Problem, Outsourcing, Insourcing

Solving Single and Multiple Plant Sourcing Problems with a Multidimensional Knapsack Model

Natalie S. Cherbaka

(ABSTRACT)

This research addresses sourcing decisions and how those decisions can affect the management of a company's assets. The study begins with a single-plant problem, in which one facility chooses, from a list of parts, which parts to bring in-house. The selection is based on maximizing the value of the selected parts, while remaining within the plant's capacity. This problem is defined as the insourcing problem and modeled as a multidimensional knapsack problem (MKP). The insourcing model is extended to address outsourcing and multiple plants. This multi-plant model, also modeled as an MKP, enables the movement of parts from one plant to another and consideration of a company-wide objective function (as opposed to a single-plant objective function as in the insourcing model).

The sourcing problem possesses characteristics that distinguish it from the standard MKP. One such characteristic is what we define as multiple attributes. To understand the multiple attribute characteristic, we compare the various dimensions in the multidimensional knapsack problem. A classification is given for an MKP as either having a single attribute (SA) or multiple attributes (MA). Mathematically, the problems of each attribute classification can be modeled in the same way with simply a different interpretation of the knapsack constraints. However, experimentation indicates that the MA-MKP is more difficult to solve than the SA-MKP. For small problems, with 100 variables and 5 constraints, the CPU time required to find the optimal solution for MA-MKP to SA-MKP problems has a ratio of 32:1.

To determine effective methods for addressing the MA-MKP, standard mixed integer programming techniques are tested. The results of this testing are that the exact approaches

are not successful in dramatically reducing the solution time to the level of the SA problems. However, a simple heuristic that performs very well on the MA-MKP is presented. The heuristic utilizes variations on the benefit-to-cost ratio and strongest surrogate constraints. The results from experimentation for MA-MKP problem sets, generated using the methods for standard MKP test data sets in the literature, are presented and indicate that the heuristic performs well and improves with larger problems. The average gap between the heuristic solution and the optimal solution is 1.39% for 200-part problems and is reduced to 0.69% when the size of the problem is increased to 298 parts.

Although the MA characteristic reflects the sourcing problem, the actual data used in the experimentation is generated with techniques presented in the literature for standard MKP test problems. Therefore, to more accurately represent the sourcing problem, industry data from a manufacturing facility is studied to identify further sourcing problem characteristics. As a result, industry-motivated data sets are generated that reflect the characteristics of industry data, yet maintain the structure of literature data sets to allow for easy comparison. It is found that both industry and industry-motivated data sets, although possessing the MA characteristic, are much easier to solve than SA problems. Indicators of difficulty appear to be the constraint tightness and a measure of the matrix sparsity. The sparsity is a significant factor because industry data tends to be very sparse, while data sets generated in the literature are completely dense. Another interesting result from the industry-motivated data sets with the single-plant problem is the tendency for a facility to prefer currently produced parts over insourcing new parts from outside the facility.

It is not uncommon for a company to have more than one facility with a particular capability. Therefore, the sourcing model is extended to include multiple facilities. With multiple-facilities, effectively all the parts are removed to form one list, and then each part is assigned to one of the facilities or outsourced externally. The multi-facility model is similar to the single-facility model with the addition of assignment constraints enforcing that each part can be assigned to only one facility. Experimentation is performed for the two-, three-, and four-facility models. The problem gets easier to solve as the number of facilities increases.

With a greater number of facilities, it is likely that for each part one of facilities will dominate as the best option. Therefore, other solutions can quickly be eliminated and the problem solved more quickly. The two-facility problem is the most difficult; however, the heuristic performs well with an average gap of 0.06% between the heuristic and optimal solutions.

We conclude with a summary on experiences with modeling and solving the sourcing problem for a sheet metal fabrication facility. The model solved for this problem had over 1857 parts with 19 machines, which translates to over 70,000 variables and 38 constraints. Although extremely large compared to problems solved in the literature, this problem was solvable because of the unique structure of industry data. Our work with the facility saved the parent organization up to $4.16M per year and provided a tool that encourages a systematic and quantitative process for evaluating decisions related to sheet metal fabrication capacity.

This work received support from the Center for High Performance Manufacturing and Ingersoll-Rand (Hussmann).

Acknowledgements

As all of life's many journeys, experiencing the growths and achievements would not be possible without the assistance and support of the people surrounding me. Reaching this significant milestone was no exception. This list is long, but writing this section was an excellent reminder, and stirred my gratefulness for the many hands that have contributed to my academic career thus far.

First and foremost, I would like to give my deepest gratitude to my graduate advisor, Dr. Russell D. Meller. This research would not have been possible without his vision and guidance. He has spent countless hours honing my skills, and ensuring that my time as a graduate student will prepare me for life's upcoming journeys.

Sincere thanks to my committee members, Dr. Ebru K. Bish, Dr. F. Frank Chen, Dr. Barbara J. Hoopes, and Dr. Hanif D. Sherali. I am grateful for their willingness to contribute to my development, and to provide constructive feedback and thought provoking questions that helped to direct this research.

I would like to thank Dr. Kimberly P. Ellis for her assistance and ideas concerning my CHPM project, Dr. Sanjay Jain for his guidance with my first teaching experience, Dr. Nancy G. Love for her hours of advice and timely stress relief on our bicycle saddles, Dr. Robert E. Taylor for teaching me racquetball and providing a listening ear combined with good advice on life, Brian Burgess and Will Vest for bailing me out when my computer "breaks," and Lynette Kessinger, Alice Clawson, Lovedia Cole, Kim Ooms, and Nicole Lafon for taking care of the details for which I would otherwise be lost.

My warmest regards to my ISE friends who have made my day to day experiences wonderful and my time worthwhile: Nathan Ivey, Qi Liu, John Klote, Amy Brown, Grace Tran, and Myrna Callison. Also, thanks to my dear friends Monica Licher, Lisa Piedl, Bob and Diana Canter, Chris Pohowski, Graham Henshaw, Dan Hibbard, The G, The FB, The T, Abe Gillespe, Brad Ball, Shawna Clark, Barrett Clark, Joe Hitchcock, and Juli Curry.

My appreciation goes out to my parents for their unwavering support, love, and interest in my life. Finally, thanks to my husband and best friend, Mark, for encouraging me to initiate this journey and seeing me through it with incredible respect and love.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The effective utilization of a company's assets is one of the key challenges facing company executives today. Asset utilization can be thought of as production of the asset divided by the asset's capacity. Assuming that an asset's capacity is fixed and production varies over time, managing the production is a critical component to addressing this challenge.

The problem addressed in this dissertation examines how sourcing decisions manage the utilization of a company's assets through varying the production at the asset. Section 1.1 discusses the motivation for this dissertation topic, a company that used insourcing as a means for managing its assets. Section 1.2 examines sourcing in general, and insourcing in particular. Finally, Section 1.3 presents how the sourcing problem is modeled using the classical knapsack problem.

1.1 Motivation

This research is motivated by work with a sheet metal manufacturing facility that currently has excess capacity. This highly automated manufacturing facility has invested a significant amount of capital in equipment that is under-utilized.

Besides the primary sheet metal facility, other divisions that are part of the parent organization require sheet metal parts. Additionally, some of these divisions outsource sheet metal parts since they do not have either the capability or capacity to produce the parts themselves. An opportunity exists to move currently outsourced sheet metal work from the other divisions to the primary facility.

To address this opportunity, the decision to source these parts (to either the primary facility or the current supplier) and the effects of these decisions on the primary facility and the parent organization, are mathematically modeled.

1.2 Sourcing

Sourcing is the process of determining where a part is manufactured; it has two forms: outsourcing and insourcing. Momme [30] defines outsourcing as “the process of entering into a contractual agreement with a supplier concerning manufacturing that so far has been provided in-house” and insourcing as “the ‘reverse’ process of outsourcing.” Insourcing would then be defined as the process of entering into a contractual agreement with a buyer concerning the manufacturing to be brought in-house. This study focuses not on the contractual agreement, but rather on the decision of whether or not to bring the manufacturing of a part in-house.

Outsourcing decisions are often referred to or included within the make-or-buy environment. The make-or-buy decision is the decision of whether to manufacture an item internally or to purchase it externally. Current make-or-buy research recognizes that outsourcing decisions should be linked to manufacturing strategy, operations, and development.

Sourcing decisions need to be made in a systematic manner; hence, frameworks have been developed to guide the decisions. A generic overview of these frameworks includes the following five common elements (some specific frameworks are addressed in Chapter 2). At each step of the framework, a multidisciplinary team should be employed to maintain the strategic overview perspective at each step.

1. Define and exploit strategic competencies, where strategic competencies are those functions that are a key source of competitive advantage.

2. Consider outsourcing non-strategic (or core) competencies.

3. Compare supplier capabilities to in-house production, both in terms of cost and performance, using well defined measures.

4. Strategically manage relationships with suppliers.

5. Re-evaluate as environments change.

These types of frameworks are created from the point of view of a company as a buyer in the sourcing relationship; that is, they provide guidance for the decision of whether or not to outsource. However, the decision to be made by a company like the primary sheet metal facility is from the point of view of the supplier in a sourcing relationship. This insourcing decision, being the reverse of the outsourcing decision, can follow a similar framework. Assuming that it follows the generic framework, our work is focused within Step 3. That is, to compare the supplier versus in-house options using well-defined measures for both cost and performance. In the sheet metal example, this translates to comparing the in-house cost (at the primary facility) to the current outsourcing costs. The performance aspect of Step 3 cannot yet be compared because the primary facility does not yet have experience acting as a supplier. This issue is being handled separately by the company.

At Step 3 in the framework, it is assumed that the sheet metal facility has the capability to build a part, or it is not considered for insourcing. For the sheet metal facility, the multidisciplinary view includes the goals of utilizing capacity and capabilities within the various divisions of the parent organization. Therefore, the performance measures and constraints used in modeling the decisions are based on both available capacity and the costs of operations in-house versus the current outsourcing costs.

1.3 Multidimensional Knapsack Approach to Sourcing

This section discusses the method used to model sourcing decisions as a multidimensional knapsack problem (MKP). This model will help answer the question that the sheet metal facility needs to answer: "If given a list of potential parts from the other divisions, which ones should be insourced? And, given the current parts produced in-house, which ones should be outsourced?" The classic knapsack problem is explained first, and then it is extended to the MKP.

The knapsack problem is a classic operations research problem concerned with filling a knapsack with a subset of available items. The knapsack in this problem has a weight capacity and each item has an associated weight and value. Assuming that all the items will not fit in the knapsack, the objective is to select the items that will maximize the total value while not exceeding the weight limit of the knapsack.

The knapsack problem represents the sourcing problem in the following way: The knapsack is analogous to the facility that has a capacity on its machine time, and the items are analogous to the parts considered for insourcing. Each part has a value to the plant and uses a specific amount of machine time. The objective is to select the parts that will maximize the total value to the plant while not exceeding the machine time capacity. Additionally, parts can be removed from the facility, which effectively increases the machine time capacity, and thus allows for more parts to be insourced.

The classic knapsack problem has a single constraint, the weight capacity of the knapsack. The multidimensional knapsack problem considers more than one constraint on the knapsack. An example of another knapsack constraint is a constraint that limits the total volume, in addition to the total weight, of the items included in the knapsack. In the sourcing problem, this corresponds to limiting both the labor time and the machine time that can be added to the plant. In addition, since each machine in the facility has a different capacity and parts are routed through more than one machine, capacity constraints are needed for each machine.

Thus, the sourcing problem is multidimensional with respect to the existence of more than one knapsack constraint. Additionally, some of the knapsack constraints represent different attributes. Therefore, the insourcing problem would be classified as a multiple-attribute MKP. These multidimensional classifications will be covered more fully in Chapter 3.

1.4 Dissertation Outline

In Chapter 2 current literature is reviewed for both sourcing decisions and the multidimensional knapsack problem (MKP). In Chapter 3 we model the sourcing problem as a multidimensional knapsack problem. Next, the multiple-attribute structure, present in the sourcing problem, is compared to the standard single-attribute structure of the multidimensional knapsack problem in Chapter 4. Because the multiple-attribute problems are difficult, in Chapter 5 we present a simple, yet effective heuristic. With an industry problem as the motivation for this research, in Chapter 6 we explore the characteristics of industry sourcing data and generate industry-motivated data sets. In Chapter 7 we extend the model to multiple facilities and present experimental results. This research is not only industry motivated, but has also been applied to a sheet metal facility; therefore, experiences with the sheet metal application are reported in Chapter 8. Finally, conclusions and future research are presented in Chapter 9.

Chapter 2

Literature Review

This chapter reviews the literature, first for the insourcing and outsourcing decisions, and then for the MKP. The closest fit to the sourcing decisions as described above is in the make-or-buy research, covered in Section 2.1. Since the sourcing decisions will be modeled as a MKP, and the MKP is an NP-hard problem, Section 2.2 focuses on efficient solution procedures for the MKP. Section 2.3 summarizes how this research relates to the current literature.

2.1 Sourcing

In the area of sourcing, the two main streams of research are the make-or-buy decisions and the operations of outsourcing. Our research does not directly fall into either area, but, as discussed in Chapter 1, it is a subset of the make-or-buy decision. Recent literature reveals that make-or-buy decisions must be made in a strategic and methodical manner. A variety of frameworks have been created that provide a sequence of steps to aid in the sourcing decisions of what, why, and how to source. Each framework has a slightly different focus and/or motivation. This study addresses some of these frameworks to understand the position of the sourcing problem within them.

Venkatesan [38] presents an approach called the strategic sourcing process that is modeled after a strategy for highly engineered products. It is composed of three main elements: focus on the strategic components, outsource when suppliers have the advantage, and outsource to narrow the focus for in-house manufacturing processes. Each subsystem, or component, is examined at its multiple stages throughout the process. A subsystem is considered strategic if it is critical for long run competitive advantages, or additionally, if it requires specialized assets or unique manufacturing design and skills. To decide if suppliers have the advantage, subsystems (strategic and non-strategic) are compared to suppliers' capabilities. For strategic subsystems, the resources required to upgrade in-house capabilities to the level of the supplier are considered. To manage the outsourcing, a supplier grading system based on performance and cost is used. The final element is to re-evaluate often.

As in Venkatesan [38], Jennings [21] also gives a broad strategic perspective. He studied several UK building societies (institutions similar to credit unions) in a time of change and determined they underestimated their use of outside supply. To develop the building societies' competitive strategies, Jennings suggests a policy with three main elements: identify and enhance strategic competencies, such as information processing and product innovation; exploit the strategic competencies through sourcing arrangements, for example, by freeing resources to be focused in strategic areas; and continually review sourcing decisions as the product and supply market changes.

Cánez *et al.* [4] attempt to narrow the previously presented broad frameworks by defining the relevant factors and providing a framework in which to evaluate these factors. The framework should possess the following characteristics: ease of understandability and use, a method of cost comparison, a definition of control in sourcing relationships, a definition of strategic capabilities, recognition of future market projections, a multi-disciplinary decision team, a generic perspective to cover a variety of industries, and the ability to be changed and updated.

Insinga and Werle [20], like Cánez *et al.* [4], also focus less on the overall picture than on how the higher level strategy affect operations. Their claim is that strategies are lost in the day-to-day operations of outsourcing. To guard against this, Insinga and Werle suggest a two-dimensional methodology. The first dimension, to define metrics that measure the value of an activity, is presented as a scale to determine the potential for competitive advantage. Beginning with the level least likely to yield a competitive advantage, the four levels of the scale are as follows: first, a readily available commodity activity; second, a basic activity needed in the business; third, an emerging activity with the potential to be a competitive differentiator; and fourth, a key activity that is currently a competitive differentiator. The second dimension is a metric to measure the performance capability of an activity in-house compared with competitors. The scale for this metric has three levels: weak, moderate, or strong. Thus, each activity has a position on a grid, and each grid provides direction on the sourcing decision.

Dekkers [8] also focuses on bridging the strategic and operational levels. The strategy is two-fold, maximizing the competitive advantage, along with the resource acquisition and utilization required for the competitive advantage. This strategy is intended to be implemented in close conjuction with manufacturing management during the early stages of product development. In contrast to Cánez *et al.* [4] and Insinga and Werle [20], Dekkers' strategy [8] is a continual process not triggered by an external influence. Since manufacturing strategy affects many process stages such as product development, manufacturing technologies, and performance requirement, continuous evaluation of requirements at each stage drives sourcing decisions.

This study assumes that the competitive strategies have already been defined, and that the capability exists to produce the parts considered for insourcing. The focus is to develop a method of comparison for in-house versus supplier production at the tactical level as oppossed to the strategic level. Therefore, the following research is a subset of the above make-or-buy frameworks and contributes toward the decisions to be made at one step of the framework.

2.2 Multidimensional Knapsack Problem

The MKP is useful for representing many problems. The traditional application is the capital budgeting problem introduced by Lorie and Savage [24]. In the capital budgeting problem projects are selected to maximize profit while not exceeding any one of the resource constraints. Gavish and Pirkul [15] modeled the allocation of processors and databases in distributed systems. MKPs have also been used to model project allocation [41], and cargo loading problems [35].

2.2.1 Exact Approaches

Exact approaches for the MKP have been developed predominantly using branch and bound, with a few approaches based on dynamic programming. Primarily the method of bound generation distinguishes each algorithm.

Balas [1] was among the first to develop an exact approach for the MKP. He presents a branch and bound approach in which all the variables start at zero and increase to one based on a systematic pseudo-dual algorithm. At each step, the algorithm identifies which branches lead to infeasible problems. The efficiency of the algorithm is dependent on the number of branches that can be eliminated. Another aspect of the algorithm's efficiency is that it does not require solving the continuous linear programming relaxation. Rather, at each step of the algorithm, only additions and subtractions are performed. This algorithm was applied to a problem with 40 variables and 22 constraints.

Soyster and Slivka [36] provide an algorithm that performs iterations of the Balas algorithm [1]. Their procedure forms subproblems using the linear programming relaxation solution, and then solves each subproblem using Balas' algorithm. The size of the subproblems is dependent on the number of constraints; hence, this algorithm performs well on problems with few constraints. They solved problems with up to 400 variables and 10 constraints.

Shih [35] presents a branch and bound procedure in which an upper bound is found by considering each of the knapsack problems independently, and then solving the relaxed linear program for each knapsack. The minimum of these knapsack bounds is the upper bound for the node in question. On problems with up to 90 variables and 5 constraints, the method was shown to outperform Balas' algorithm [1] with respect to both solution time and number of iterations.

Gavish and Pirkul [15] develop and compare the bounds obtained by relaxations of the MKP. They develop Lagrangean, surrogate (aggregation of all knapsack constraints into one), and composite (combination of surrogate and Lagrangean) relaxations. In problems with up to 300 variables and 5 constraints, or 500 variables and 3 constraints, their branch and bound procedure outperforms Shih's algorithm [35] in both CPU time and the size of solvable problems.

Gabrel and Minoux [13] present a scheme to identify the most violated extended covers inequalities. The violated inequalities are those that are valid to the MKP, yet violated by the linear relaxation solution. They use a ratio between the left- and right-hand sides to measure constraint violation and to generate minimal covers (a necessary condition for the inequality to be a facet). They show a reduction in CPU time as compared to the standard CPLEX MIP solver on problems with up to 180 variables and 60 constraints.

Gilmore and Gomory [16] present a modified dynamic programming (DP) algorithm using single dimensional knapsack problem characteristics. They derive a divide-in-two inequality from the single dimensional cutting stock problem: $F(x_1 + x_2) \geq F(x_1) + F(x_2)$, where x_1 and x_2 are the length of each item, and $F(x)$ is the knapsack objective function value. This divide-in-two inequality extended to the two dimensional problem is used in the dynamic programming fundamental forward recursion equation.

Weingartner and Ness [42] develop a DP approach for the basic capital allocation problem that includes various ordering schemes, use of the complement problem, and upper bounds found by solving the relaxed linear program. They employ a simple scheme that is useful when the constraints are loose. At each stage of the scheme, the remaining items are checked for addition into the knapsack without violating a constraint. If feasible, the solution obtained is also a lower bound. They solve problems with 2 constraints and up to 105 variables.

Nemhauser and Ullmann [31] extend the work of Weingartner and Ness [42] on a DP approach to the capital allocation problem. The extensions include multi-level projects or projects accepted at varying levels of investment and return; reinvesting returns, potentially creating negative coefficients on the constraints; borrowing; deferral of capital until later periods; and most notably, incorporating dependent or interacting projects, where the acceptance of one project is dependent on the returns of another project or projects share equipment. With interacting projects, the objective function becomes non-linear, and the new algorithm is based on DP for non-serial systems.

The MKP is well known to be NP-complete [14], and thus the size of problems that can be solved optimally is limited. In the above research, problems are solved optimally up to about 400 variables and 10 constraints. Discussed in Chapter 3, the Hussmann sheet metal insourcing problem requires up to 12,000 variables and 36 constraints. Therefore, heuristics that can solve larger problems are of particular interest.

2.2.2 Heuristic Approaches

Primal heuristics have been used to solve problems with up to about 1,000 variables and 20 constraints. Most primal heuristics either begin with a solution where no items are included in the knapsack, and items are added one at a time, based on a given rule, while maintaining feasibility, or they begin with all items included and then removed one at a time until the solution is feasible.

Toyoda's approach [37] begins with a feasible solution and all variables equal to zero, and then adds items one at a time based on a preferability ranking of the variables. The preferability measure is calculated using an effective gradient with a penalty factor. The penalty vector is formed by the penalties associated with each resource constraint, where an individual constraint penalty is relative to the total amount that all the items require of each respective resource. Additionally, each item has a necessary resource vector containing the amount of each resource that the item requires. The length of the necessary resource vector, when projected on the penalty vector, is an element of the preferability measure. The preferability measure is a ratio of the value of an item over the projected length of the necessary resource vector. Problems are solved with up to 1000 variables and 1000 constraints.

Loulou and Michaelides [25] use a similar idea by choosing the item to enter next with the maximum pseudo-utility factor. As with the preferability measure by Toyoda [37], an item's pseudo-utility factor depends on its profit and resource consumption. The penalty factor for each item (different from Toyoda's [37]) is a function of the total resource consumption of the item, the remaining resources after the item is selected, and the potential demand for each resource after the item is selected. The pseudo-utility factor, used to choose the entering item, is then the profit of an item divided by the penalty factor. This heuristic performed slightly better than Toyoda's method [37] with respect to solution quality.

The next type of heuristic focuses on bound calculations to drive the heuristics. Heuristics in this category have solved problems in the literature with up to about 1,000 variables and 20 constraints, or 20 variables and 1,000 constraints.

Balas and Martin [2] were among the first to use bound calculations to drive their heuristic, called Pivot and Complement. In the first stage, Pivot and Complement uses linear programming to calculate an upper bound and then heuristically sets the non-integer solution to integer. A series of pivots moves slack variables into the basis. The second stage is an improvement procedure that complements variables while maintaining primal feasibility. They solved problems with up to 900 variables and 200 constraints. This heuristic outperformed Toyoda's method [37] with respect to solution quality, but at the cost of about twice the CPU time.

Magazine and Oguz [27] developed an algorithm, Multi-Knap, that combines the dual heuristic method of Senju and Toyoda [34] with Everett's Generalized Lagrange Multipliers (GLMs) approach [10]. As in Senju and Toyoda [34], Multi-Knap begins with the relaxed solution with all variables equal to one and all GLMs at zero. The GLMs are adjusted one variable at a time until the solution is primal feasible. Magazine and Oguz solved problems with up to 1000 variables with 20 constraints and up to 20 variables with 1000 constraints. Multi-Knap performs similarly to the Senju and Toyoda heuristic in terms of CPU time, but has slightly improved solution quality. The complexity of Multi-Knap is shown to be $O(mn^2)$, where n is the number of variables, and m the number of constraints.

Volgenant and Zoon [40] improve on Magazine and Oguz's Multi-Knap [27] by computing the GLMs simultaneously as opposed to stepwise in Multi-Knap. Volgenant and Zoon also present an upper bound improvement at the end of the heuristic by changing some multiplier values. With the new bound, the complexity of this heuristic is $O(n(n+m))$. The heuristic was tested on randomly generated problems with up to 200 variables and 200 constraints with varying constraint slackness. The heuristic was also tested on the problems from Senju and Toyoda [34]. On average, Volgenant and Zoon's algorithm was better than Multi-Knap with respect to solution quality, but worse with respect to CPU time.

The heuristic by Lee and Guignard [23] uses a modification of Toyoda's method [37]. In their approach, Lee and Guignard set more than one variable at a time to find a feasible solution in the first phase. The second phase then improves the solution with a modification of the complementing procedure used by Balas and Martin [2]. The second phase also identifies the number of variables to be complemented by problem instance characteristics. This algorithm was tested on problems with up to 500 variables and 30 constraints. The problems were both randomly generated and taken from the literature (Senju and Toyoda [34], Jeroslow and Smith [22], and Balas and Martin [2]). Compared to the Balas and Martin approach, on average, Lee and Guignard's approach is much better with respect to CPU time, but worse with respect to solution quality.

Bertsimas and Demir [3] use an approximate dynamic programming approach. They approximate the value function using a base-heuristic approach and an adaptive fixing heuristic. The base-heuristic approach estimates the optimal value function by constructing a sub-optimal solution to a subproblem. Some of the variables in each subproblem are assigned values based on reduced costs, and the other variables are iteratively assigned using dynamic programming techniques with an approximate value function. The adaptive fixing heuristic solves linear programming relaxations iteratively and uses those solutions to fix variables. Bertsimas and Demir solved problems with up to 1000 variables and 100 constraints. Compared to the commercial package CPLEX 6.0 [6], on average, the algorithm competes with, and often out-performs CPLEX in terms of CPU time.

More recently, metaheuristics (a general structure for heuristics to solve hard problems using a global search) have been developed to solve the MKP on problems of similar size to the bound-based heuristics, but with improved solution quality. The specific heuristic developed requires the definition of parameters and decision variable representation. Tabu search, genetic algorithms, and simulated annealing are some of the most common metaheuristics.

Tabu search is based on adaptive memory structures and a responsive exploration of the solution space. The memory structures maintain and update a list of visited solutions and features of those solutions. Solutions on the tabu list are to be avoided. The responsive exploration allows the good solution features to be exploited. Some tabu search development issues are identifying which attribute to trace, defining the tabu duration, and defining the aspiration criteria that allows the overriding of the tabu list.

A genetic algorithm is inspired by the field of genetics and the development of a population. A solution is represented by a member of the population and the search is driven by reproduction, mutation, and crossover evolution of a population.

Simulated annealing is inspired by the process of annealing metal in which a metal is slowly cooled until reaching a minimum energy state. The heuristic allows the search to move to a non-improving solution with a probability that decreases with time, according to a cooling schedule.

Many of these metaheuristics use a pseudo-utility function to drive the search. This function corresponds to the value to weight ratio in the single constraint 0-1 KP. Most of the metaheuristics are tested on 57 standard literature problems used in Freville and Plateau [12] (made available by Chu and Beasley [5] in the OR-library [32]) with 6 to 105 variables and 2 to 30 constraints, and on 24 benchmark problems presented by Glover and Kochenberger [17], with 100 to 500 variables and 15 to 25 constraints, that are known to be difficult to solve for branch and bound algorithms.

Dammeyer and Voß [7] present a tabu search with a dynamic tabu list (tabu duration is not constant) where the tabu duration is determined according to the solution attributes using the reverse elimination method. This method allows a solution only to be re-visited in the next iteration if it is a neighbor of the solution at the current iteration. A move to a new solution is made by dropping one variable, or assigning it to zero, and adding one or more variables, or increasing them to one, while maintaining feasibility. This method is tested on the 57 problems in Freville and Plateau [12]. This tabu search outperformed Drexl's [9] simulated annealing with respect to the number of problems solved to optimality, the average deviation from the optimal solution, average CPU time, and the average number of moves.

Hanafi and Freville [18] employ a tabu search in which they oscillate between feasible and infeasible solutions, as opposed to Dammeyer and Voß [7] where feasibility is maintained. The oscillation strategy is defined by the surrogate constraints; i.e., constraints in which multiple constraints are joined into one. Hanafi and Freville use a greedy search to intensify the search within a promising zone, and, to diversify, the search moves away from the promising zone into either feasible or infeasible solutions. The optimal solution was found in all the standard problem instances from Freville and Plateau [12] and Glover and Kochenberger [17]. Additionally, the method outperforms Glover and Kochenberger [17] with respect to CPU time.

Vasquez and Hao [39] present a hybrid approach with tabu search and linear programming

(LP). They use the LP relaxation to define the search area and then tabu search to intensify the search. This heuristic also finds the optimal solution to the standard test problems mentioned above. Compared to another set of problems in the literature, from the OR-library proposed by Chu and Beasley [5], this heuristic improves on the pervious performances in each measure. Finally, it also gives an improved solution for 9 of the 11 instances of more recent standard problems presented by Glover and Kochenberger [17], with up to 2500 variables and 100 constraints.

Chu and Beasley [5] present a genetic algorithm that considers MKP specific knowledge and maintains solution feasibility using a greedy repair heuristic. Parents are selected by choosing the most fit parent from each of two randomly formed pools. For crossover and mutation, a simple uniform crossover is implemented, where a bit is chosen randomly from one of the parents and a few bits are mutated after the crossover. This approach found the optimal solution to each of the OR-library problems and outperformed Magazine and Oguz [27] and Volgenant and Zoon [40] in terms of solution quality.

2.2.3 Multidimensional Knapsack Problem Summary

Although there are other algorithms and heuristics for the MKP, the above approaches summarize the literature in terms of breadth and performance. For further details on the status of the MKP, see the recent survey by Fréville [11].

Two characteristics to make note of in the above literature are the constraint generation and the size of the problems solved. The first characteristic present in all of the experimental problems solved in the literature is that, for each problem, the constraints all represent the same attribute, and the problems can be classified as single-attribute MKPs. As will be described in Chapter 3, the sourcing problem is formulated with multiple attributes across the constraints. Second, the size of the problems solved in the literature is much smaller than the Hussmann sourcing problem size, with up to 12,000 variables and 36 constraints. Table 2.1 provides a summary of various approaches, and the size of the problems solved by each method.

Table 2.1: Size of MKP Problems Solved

	Variable Range		Constraint Range	
Authors	**Min**	**Max**	**Min**	**Max**
Exact Approaches				
Gilmore and Gomory [16]		20		2
Balas [1]		40		22
Shih [35]	30	90		5
Weingartner and Ness [42]		105		2
Gabel and Minoux [13]		180		60
Soyster and Slivka [36]	50	400	5	10
Gavish and Pirkul [15]	20	500	3	5
Heuristic Approaches				
Senju and Toyoda [34]		60		30
Bertsimas and Demir [3]	20	105	2	30
Dammeyer and Voss [7]	20	105	2	30
LouLou and Michaelides [25]	20	105	10	330
Volgenant and Zoon [40]	25	200	25	200
Hanafi and Fréville [18]	20	500	2	30
Chu and Beasley [5]	20	500	2	30
Lee and Guignard [23]	60	500	5	30
Balas and Martin [2]	20	900	5	200
Magazine and Oguz [27]	20	1000	20	1000
Toyoda [37]	50	1000	50	1000
Vasquez and Hao [39]	20	2500	2	100

2.3 Summary

A distinguishing characteristic of the sourcing frameworks in the literature is that the company is viewed as the buyer in the sourcing relationships; that is, they decide whether or not to outsource. This study focuses more on the decision of whether or not to insource (with outsourcing viewed as a way to free capacity for further insourcing). Additionally, this research assumes that the strategic and core competencies are defined and that eligible parts for outsourcing or insourcing are known. Current sourcing literature focuses on the strategic level decisions and how they interact with tactical decisions. Therefore, the decisions that will be modeled are a subset of the current make-or-buy decision environment in that the focus is on the tactical level decisions. The question we want to answer is, given a list of parts that can potentially be insourced, which parts should be selected based on the production costs and capacities. No framework currently exists to address this insourcing problem as defined by the Hussmann example.

Presented in Chapter 3, this sourcing problem can be modeled as a MKP with the characteristic that each knapsack constraint represents a different physical constraint (e.g., machine time, labor time, multiple machines, etc). Current literature solves the MKP optimally for up to 400 variables and 10 constraints. However, the Hussmann sourcing problem can be up to three times that size. Furthermore, the current multidimensional knapsack research experiments with standard data sets that do not reflect the sourcing problem. Literature data sets assume that each knapsack constraint represents the same attribute (or each knapsack constraint is sampled from the same distribution). However, with this sourcing problem, each knapsack constraint is potentially modeled with respect to a different attribute. Therefore, Chapter 4 focuses on the effect of multiple attributes when solving multidimensional knapsack problems. Other characteristics of industry data not present in the literature data sets are identified and addressed in Chapter 6.

Chapter 3

Problem Statement

This chapter describes the relationship of the sourcing problem to the Multidimensional Knapsack Problem (MKP). Section 3.1 defines the MKP in terms of the Knapsack Problem (KP) and the ways it can be extended into the MKP. Additionally, the different types of knapsack constraints are compared in both structure and interpretation. In Section 3.2 the general insourcing problem is modeled as an MKP. Then, variations of the insourcing problem are presented in Section 3.3. These variations include the addition of outsourcing (or the sourcing problem), consideration of time periods, and a specific model for the sheet metal example.

3.1 Multidimensional Knapsack Problem

3.1.1 Knapsack Problem

The 0-1 MKP is a generalization of the 0-1 KP; therefore, discussion begins with the KP and ways to extend into an MKP. In the KP, there exist n items and one knapsack of capacity b. Each item, i, has a weight, w_i, and value (or profit), p_i. The decisions are whether or not to include each item in the knapsack. These decisions are represented in the KP by the binary variables $\mathbf{x}$, where x_i is equal to one if item i is included in the knapsack, and zero otherwise. The objective is to fill the knapsack with the items that maximize the value of the selected items while remaining within the knapsack's capacity. The KP formulation follows:

$$\begin{aligned} \text{Maximize} \quad & \sum_{i=1}^{n} p_i x_i \\ \text{subject to} \quad & \sum_{i=1}^{n} w_i x_i \leq b \\ & x_i \in \{0,1\} \qquad \forall i = 1, 2, \ldots, n \end{aligned}$$

3.1.2 Multiple Attributes

To extend the KP to include multiple dimensions, the first step is to add an additional attribute, such as volume, to the above one-dimensional problem. This problem will be referred to as the two-dimensional knapsack problem (KP-2D). The knapsack has both a weight capacity, b^w, and a volume capacity, b^v. Letting v_i be the volume of item i, the formulation follows:

$$\begin{aligned} \text{Maximize} \quad & \sum_{i=1}^{n} p_i x_i \\ \text{subject to} \quad & \sum_{i=1}^{n} w_i x_i \leq b^w \\ & \sum_{i=1}^{n} v_i x_i \leq b^v \\ & x_i \in \{0,1\} \qquad \forall i = 1, 2, \ldots, n \end{aligned}$$

Looking at the previous two formulations, the difference between the KP-2D and the KP is simply one additional constraint, referred to as a knapsack constraint. The existence of more than one knapsack constraint implies a multidimensional knapsack problem (MKP).

3.1.3 Multiple Knapsacks

The KP-2D problem above represents one type of MKP in that the knapsack constraints represent two attributes, volume and weight. A different type of MKP could be (starting with the KP) to add an additional knapsack with the requirement that if an item is included in one knapsack, it must also be in the other. However, each item's weight can differ depending on the knapsack. This can be thought of as two subparts of an item that in general have different weights and must be kept in separate knapsacks. If either subpart of an item is included, the other must also be included in the other knapsack. This problem is a knapsack problem with one attribute and two knapsacks (2KP). The formulation follows, where w_{i1} and w_{i2} are the weight of item i in knapsack one and two, respectively, and b_j^w is the weight capacity of knapsack j, $j = 1, 2$.

$$\begin{array}{lll}
\text{Maximize} & \displaystyle\sum_{i=1}^{n} p_i x_i & \\
\text{subject to} & \displaystyle\sum_{i=1}^{n} w_{i1} x_i \leq b_1^w & \\
& \displaystyle\sum_{i=1}^{n} w_{i2} x_i \leq b_2^w & \\
& x_i \in \{0, 1\} & \forall i = 1, 2, \ldots, n
\end{array}$$

Comparing the two previous problem formulations, KP-2D and 2KP, the two problems have different interpretations, but both formulations represent a knapsack problem with one additional knapsack constraint. Mathematically, problems KP-2D and 2KP are equivalent.

These problems are both generally referred to as a bi-dimensional knapsack problem, regardless of the number of attributes the constraints represent. It is possible that, although the formulations are mathematically equivalent, problem characteristics may exist for each type of formulation that can be utilized in a solution procedure, or that may influence the computational complexity. Therefore, this research differentiates between the two types of multidimensional knapsack interpretations, single-attribute MKP and multiple-attribute

MKP.

3.1.4 Multidimensional Knapsacks

In the most general case, the MKP has m knapsacks (or m-dimensions), each of capacity b_j, n items, and the weight (or coefficient) of each item can be different in each knapsack. Similar to the KP, the objective is to find the items that maximize the value of the knapsacks while not exceeding the capacity of any one knapsack. The MKP mathematical formulation follows, where w_{ij} is the weight of item i in knapsack j, and b_j is the capacity of knapsack j.

$$
\begin{aligned}
\text{Maximize} \quad & \sum_{i=1}^{n} p_i x_i & & \\
\text{subject to} \quad & \sum_{i=1}^{n} w_{ij} x_i \le b_j & \quad & \forall j = 1, 2, \ldots, m \\
& x_i \in \{0, 1\} & & \forall i = 1, 2, \ldots, n
\end{aligned}
\tag{3.1}
$$

This problem is m-dimensional, and, therefore, multidimensional because m knapsack constraints are represented by (3.1). Because the knapsack constraints can be modeled similarly regardless of the number of attributes represented, the MKP can be used to model a variety of problem aspects. For example, Mansini and Speranza [28] build an MKP for asset-backed securitization where multiple knapsacks are used to model discretized time. The objective is to select a set of assets to minimize the gap between the outstanding principal of the loan and the sum of the assets in each time period. In contrast to the previous examples where the constraints represent the attributes of volume or weight, in this problem, the constraints represent time periods.

The next section describes the application of the MKP to an insourcing model in which both classifications of the MKP are utilized (i.e., single- and multiple-attribute MKPs). Using multiple knapsacks to represent time, as shown in [28], can also be used to model seasonality, with a set of multiple knapsacks, each with multiple attributes, for each time period (e.g., season, month, etc.).

3.2 Insourcing Model

In the sheet metal problem as described in Chapter 1, the primary facility has excess capacity, while other divisions of the company are currently outsourcing parts that could be manufactured in this plant. The insourcing problem is defined as, which, if any, of these parts should be insourced into the primary facility, given the available capacity, in order to maximize the value of the selected parts. The value of a part is a combination of savings to the other divisions and profit for the under-utilized plant.

Consider a part to be processed on a single machine. Two attributes, machine time and labor time, represent "weights," with the machine being constrained by both types of weights. Therefore, multiple attributes, such as those in formulation KP-2D, are used to model both the machine and labor time that a part adds to the machine. Next, assume the part requires processing on multiple machines. Multiple knapsacks are used, such as in formulation 2KP, to represent the multiple machines (or departments). A part is analogous to an item, where a part is defined by its routing through the facility. A part takes up time (both machine and labor, which can be different) at each machine and has a certain amount of value if insourced. Value is defined as the difference between the in-house production costs and the current cost of outsourcing the part, and has both a savings component and a profit component. For example, assume that a part is currently outsourced for \$30, but can be produced for \$10 by the insourcing plant, which then sets the selling price at \$15. In this scenario, the profit is \$5 and the savings is \$15, which determines the value as \$20. This value is what defines p_i in the formulation below; therefore, in this case, p_i would equal \$20. A more thorough discussion of "value" will be discussed later in Chapter 8 with the presentation of the results from of our work in industry.

The machine time capacity is defined as the time available on each machine after the current plant load is processed. Similarly, the labor time capacity is defined as the labor time available at each machine after the current plant load is processed. The decision is, then, which of the n parts to choose to maximize the value of the selected parts and remain within the available capacity of the m machines in the facility. The mathematical formulation follows, where t_{ij}^m is the *machine* time part i uses on machine j, t_{ij}^ℓ is the *labor* time part i uses on machine j, b_j^m is the *machine* time capacity on machine j, and b_j^ℓ is the *labor* time capacity on machine j.

$$
\begin{aligned}
\text{Maximize} \quad & \sum_{i=1}^{n} p_i x_i & & \\
\text{subject to} \quad & \sum_{i=1}^{n} t_{ij}^m x_i \le b_j^m & \forall j = 1, 2, \ldots, m & \qquad (3.2)\\
& \sum_{i=1}^{n} t_{ij}^\ell x_i \le b_j^\ell & \forall j = 1, 2, \ldots, m & \qquad (3.3)\\
& x_i \in \{0, 1\} & \forall i = 1, 2, \ldots, n &
\end{aligned}
$$

The above knapsack formulation has a dimension of $2m$ as there are m knapsacks for each of the two attributes, machine time and labor time. Each of the first m knapsack constraints, (3.2), limits the parts selected for insource on each machine by its available machine time; and the second set of m knapsack constraints, (3.3), limits the parts selected for insource on each machine by its available labor time. Since more than one attribute is represented in this problem, it is considered a multiple-attribute MKP.

Labor time and machine time need to be separate constraints for two reasons. First, labor time may be restricted by the number of shifts staffed, where some machines are available even when they are not staffed. In this scenario, a facility might be interested in adding additional labor beyond the current schedule. Second, machine time and labor time are not always the same for a part because processing of some parts may require more than one laborer, or labor supervision on only an occasional basis, depending on the complexity of the machine or the part. Extensions such as these are discussed next.

3.3 Extensions to the Insourcing Model

In this section two extensions to the insourcing model presented in Section 3.2 are discussed. Section 3.3.1 extends the insourcing model to consider time periods. Section 3.3.2 extends the insourcing model to consider a relaxation of the labor constraint.

3.3.1 Insourcing Model with Time Periods

As suggested earlier, multiple sets of knapsacks can be used to model a time attribute. This section extends the insourcing formulation to include time periods, enabling the modeling of seasonality effects. For example, although the capacity of machines is often greater during peak time periods when extra shifts and/or overtime are used, the available capacity at a machine is, in general, much smaller because the machines are heavily utilized by the current plant load. In this case, the b_j^m and b_j^ℓ values vary by time period and, thus, in the formulation noted as b_{js}^m and b_{js}^ℓ; the machine and labor time capacity, respectively, for machine j in season s, where $s = 1, 2, \ldots, S$, and S is the number of seasons (or time periods) considered. The time a part takes at specific machines can also vary depending on the load of the facility. Therefore, in this formulation, t_{ijs}^m and t_{ijs}^ℓ are the machine and labor time, respectively, required to process the entire quantity of part i on machine j in time period s. The formulation follows:

$$
\begin{array}{lll}
\text{Maximize} & \sum_{i=1}^{n} p_i x_i & \\
\text{subject to} & \sum_{i=1}^{n} t_{ijs}^m x_i \leq b_{js}^m & \forall j = 1, 2, \ldots, m, \quad \forall s = 1, 2, \ldots, S \\
& \sum_{i=1}^{n} t_{ijs}^\ell x_i \leq b_{js}^\ell & \forall j = 1, 2, \ldots, m, \quad \forall s = 1, 2, \ldots, S \\
& x_i \in \{0, 1\} & \forall i = 1, 2, \ldots, n
\end{array}
$$

In this case the dimension of the model increases to $2mS$ since each machine during each season requires a knapsack constraint for both machine and labor time.

The above formulation assumes that if a part is chosen, then it is added to the plant load in every time period. This is often the case as a part will be insourced only if it can be produced to meet demand in every season. A possible relaxation is to allow a part to be chosen in each time period independently of the other time periods; that is, relax x_i to x_{is}, where x_{is} is a binary decision variable to add part i in time period s. This new problem can then be separated into multiple problems by period.

3.3.2 Sheet Metal Insourcing Model

The next formulation is an application of the insourcing model (without time periods). In this formulation, the labor time capacity is no longer treated as a hard constraint. It is assumed that additional labor, above what is available, can be purchased at a known labor rate. The labor time capacity is represented by f_j, defined as the amount of "free" labor available on machine j. This refers to the labor time already allocated and paid for at each machine. It is assumed that only labor hours used beyond f_j, denoted by the decision variable h_j, are charged at the labor rate. Additionally, labor time is continuous; that is, any portion of hours or workers can be added. The notation for the decision variables and parameters follows:

Decision variables:

- $x_i = 1$ if part i is chosen, 0 otherwise, $\forall i = 1, 2, \ldots, n$
- h_j = charged labor time, time used above the available (free) labor time, $\forall j = 1, 2, \ldots, m$

Parameters:

- p_i = value/profit for insourcing part i, $\forall i$

- L = labor rate (assumed to be positive; i.e. $L > 0$)
- t^m_{ij} = machine time used by part i on machine j, $\forall i, j$
- t^ℓ_{ij} = labor time used by part i on machine j, $\forall i, j$
- b^m_j = available machine time on machine j, $\forall j$
- b^ℓ_j = maximum available labor time on machine j, $\forall j$
- f_j = free labor time available at machine j, $\forall j$

Using the above notation, the sheet metal insourcing model is presented as follows:

$$
\begin{array}{llll}
(P_{sm}) \quad \text{Maximize} & \sum_{i=1}^{n} p_i x_i - L \sum_{j=1}^{m} h_j & & \\
\text{subject to} & \sum_{i=1}^{n} t^m_{ij} x_i \le b^m_j & \forall j = 1, 2, \ldots, m & (3.4) \\
& h_j \ge \sum_{i=1}^{n} t^\ell_{ij} x_i - f_j & \forall j = 1, 2, \ldots, m & (3.5) \\
& 0 \le h_j \le b^\ell_j - f_j & \forall j = 1, 2, \ldots, m. & (3.6) \\
& x_i \in \{0, 1\} & \forall i = 1, 2, \ldots, n &
\end{array}
$$

In this model, the objective is to maximize value minus additional labor costs. As stated above, labor costs are charged only for the time used above the free labor hours. The next remark shows that the above formulation correctly models this relationship.

Remark 1 *At optimality in Problem 3.4,* $h_j = \max\{\sum_{i=1}^{n} t^\ell_{ij} x_i - f_j, 0\}$.

Proof: Given $h_j \ge \sum_{i=1}^{n} t^\ell_{ij} x_i - f_j$ and $h_j \ge 0$, $\Rightarrow h_j \ge \max\{\sum_{i=1}^{n} t^\ell_{ij} x_i - f_j, 0\}$. Given $L > 0$ and the objective is to maximize $\sum_{i=1}^{n} p_i x_i - L \sum_{j=1}^{m} h_j$, $\Rightarrow$ the model will minimize each value of h_j. Minimizing each h_j and $h_j \ge \max\{\sum_{i=1}^{n} t^\ell_{ij} x_i - f_j, 0\} \Rightarrow h_j = \max\{\sum_{i=1}^{n} t^\ell_{ij} x_i - f_j, 0\}$. ∎

As shown above, all available free labor is consumed before adding labor to produce the insourced parts. Similar to the other formulations, machine time used for the added parts cannot exceed the available machine time as stated in (3.4). Although labor hours can be added at a cost, the amount that can be added before the machine becomes the bottleneck is limited. This maximum on the total labor at a machine is specified by constraints (3.5) and (3.6), where the sum of the free labor time and charged labor time cannot exceed the available labor time.

Two possible extensions to this model are relaxing the binary constraint on the x_i values and adding seasonality. Relaxing the binary constraint on the x_i values implies allowing x_i to take on values in the range between zero and one. Then, the problem parameters account for processing the full required quantity of a part. In previous formulations, either the entire quantity would be selected or nothing. With the binary relaxation, the x_i value represents the portion of the full quantity to be selected. Adding seasonality can be handled with the method discussed in Section 3.3.1 for time periods. The resulting formulation for the sheet metal insourcing model with time periods and partial quantities follows:

$$
\begin{array}{lcl}
\text{Maximize} & \sum_{i=1}^{n}\sum_{s=1}^{S} p_{is}x_i - L\sum_{j=1}^{m}\sum_{s=1}^{S} h_{js} & \\
\text{subject to} & \sum_{i=1}^{n} t_{ijs}^{m}x_i \leq b_{js}^{m} & \forall j=1,2,\ldots,m,\ \forall s=1,2,\ldots,S \\
& h_{js} \geq \sum_{i=1}^{n} t_{ijs}^{\ell}x_{is} - f_{js} & \forall j=1,2,\ldots,m,\ \forall s=1,2,\ldots,S \\
& 0 \leq h_{js} \leq b_{js}^{\ell} - f_{js} & \forall j=1,2,\ldots,m,\ \forall s=1,2,\ldots,S \\
& 0 \leq x_i \leq 1 & \forall i=1,2,\ldots,n,\ \forall s=1,2,\ldots,S
\end{array}
$$

3.3.3 Sourcing Model

Similar to the addition of labor, the addition of an outsourcing option effectively provides an opportunity to increase the capacity of the knapsack, or rather free up machine and labor capacity that is currently consumed. In the insourcing model, the right-hand-side value represents the available capacity after the current plant load is processed. Adding outsourcing to the model is an attempt to capture the situation in which parts are considered for outsourcing in order to free capacity for more profitable parts to be insourced. In the following formulation, the insourcing decision variable, x_i, is the same as in the insourcing models, that is, it is equal to one if item i is selected for insourcing, and zero otherwise. The outsourcing decision variable, y_h, is equal to one if item h is selected for outsourcing, where $h = 1, \ldots, k$ and k is the number of parts that can be outsourced. The value of a part is p_i for insourced part i and p_h for outsourced part h. The resulting formulation follows:

$$
\begin{aligned}
\text{Maximize} \quad & \sum_{i=1}^{n} p_i x_i + \sum_{h=1}^{k} p_h y_h & & \\
\text{subject to} \quad & \sum_{i=1}^{n} t_{ij}^{m} x_i - \sum_{h=1}^{k} t_{hj}^{m} y_h \leq b_j^{m} & \forall j = 1, \ldots, m & \qquad (3.7) \\
& \sum_{i=1}^{n} t_{ij}^{\ell} x_i - \sum_{h=1}^{k} t_{hj}^{\ell} y_h \leq b_j^{\ell} & \forall j = 1, \ldots, m & \qquad (3.8) \\
& x_i \in \{0, 1\} & \forall i = 1, \ldots, n & \\
& y_h \in \{0, 1\} & \forall h = 1, \ldots, k &
\end{aligned}
$$

Constraint sets (3.7) and (3.8) represent that the amount of time used by the parts selected for insourcing, minus the time freed by parts selected for outsourcing, must remain less than the available capacity. This problem effectively chooses the parts to outsource when either it is profitable to outsource, or when it is more profitable to use the capacity on an insourced part. However, note that the constraint coefficients associated with the outsourcing variables, t_{hj}^{m} and t_{hj}^{ℓ}, are all negative. The next remark addresses this issue.

Remark 2 *The outsourcing formulation can be transformed so that all the coefficients are positive.*

Proof: Case 1: When $p_h > 0$, if $y_h = 1$ the objective function will increase and extra capacity will be available in the knapsack constraints. Therefore, if $p_h \geq 0$, y_h will always equal one and those variables can be removed from the problem.

Case 2: When $p_h < 0$, since $t^m_{hj} < 0$ and $t^\ell_{hj} < 0$ for all j, substituting $1 - y'_h = y_h$ will allow all the coefficients to be positive. ∎

The resulting formulation follows:

$$
\begin{array}{lll}
\text{Maximize} & \sum_{i=1}^{n} p_i x_i + \sum_{h=1}^{k} p_h \left(1 - y'_h\right) & \\
\text{subject to} & \sum_{i=1}^{n} t^m_{ij} x_i + \sum_{h=1}^{k} t^m_{hj} y'_h \leq b^m_j + \sum_{h=1}^{k} t^m_{hj} & \forall j = 1, \ldots, m \\
 & \sum_{i=1}^{n} t^\ell_{ij} x_i + \sum_{h=1}^{k} t^\ell_{hj} y'_h \leq b^\ell_j + \sum_{h=1}^{k} t^\ell_{hj} & \forall j = 1, \ldots, m \\
 & x_i \in \{0, 1\} & \forall i = 1, \ldots, n \\
 & y'_h \in \{0, 1\} & \forall h = 1, \ldots, k
\end{array}
$$

Previous models assumed that the current plant load was constant and the only decision was whether or not to bring each part on the list of potential parts that can be insourced. In this model, with the outsourcing of each current plant load part as an additional decision, effectively the current plant load is emptied and each part is added to the list of potential parts that can be insourced. Then, the decision becomes whether or not to insource the parts from a list containing both the original parts considered for insourcing and the parts considered for outsourcing. Finally, a part from the plant load is outsourced if it is not selected to be brought back in-house via insourcing.

This model with both insourcing and outsourcing decisions represented will be referred to as the sourcing model. Note that both the sourcing and insourcing models are standard MKPs. Therefore, the standard MKP data sets used in the experimentation can be interpreted as either insourcing-only or sourcing data sets. This distinction will be addressed more fully in Chapter 6. But first, in Chapter 4 we explore the differences in solution difficutly between MKPs with the multiple-attribute structure present in sourcing problems and the standard single-attribute structure.

Chapter 4

Multiple-Attributes of the MKP

In this chapter we are concerned with the multiple-attribute MKP in general, and specifically, the way that the multiple-attribute structure affects the MKP formulation for the sourcing problem. A multiple-attribute MKP (MA-MKP) for this research is defined to be a MKP in which at least some of the constraints are sampled from different distributions prior to any scaling. Similarly, a single-attribute MKP (1A-MKP) is a MKP in which all the constraints have the same scale, or they are generated from the same distribution.

4.1 Multiple-Attributes Versus Single-Attributes

The difference between a MA-MKP and a 1A-MKP is best explained by recalling Sections 3.1.2 and 3.1.3 in which we discuss the difference between multiple attributes and multiple knapsacks. In Section 3.1.2 the multiple-attribute MKP model (volume and weight constraints) is presented, and in Section 3.1.3 is the 1A-MKP multiple knapsacks model (two weight constraints).

The MA-MKP version of the problem is of particular interest since the constraints in the sourcing problem represent multiple attributes. The constraints in the sourcing problem at a minimum are associated with different problem features, machine time and labor time for each of the different machines. Additionally, each part is likely to require a variation of processing times across the machines on the part's routing. Therefore, because the coefficients associated with each constraint can look very different, the sourcing problem is labeled as an MA-MKP.

The literature does not consider the differences between the knapsack constraints of the MKP. All of the problems used to test the various algorithms are 1A-MKP type problems. That is, all the constraints are sampled from the same distribution. We conjecture that it is more difficult to solve a MA-MKP than a 1A-MKP. If the MA-MKP is more difficult, then there are research issues involved with identifying problem characteristics that will lead to algorithms specifically designed for a MA-MKP.

We want to answer the question of whether it is more difficult to solve a MA-MKP than a 1A-MKP. Since the two types of problems are mathematically equivalent (as shown in Section 3.1.3), the comparison between the two types of problems will be examined empirically. Problems can be generated that are the same size in terms of the number of variables and constraints, but differ in terms of the number of attributes represented by the knapsack constraints. That is, the problems differ in that one is a MA-MKP and one is a 1A-MKP. Testing has been performed using the CPLEX MIP solver [6].

We conjecture that the MA-MKP is more difficult to solve than the 1A-MKP. Problems are generated using the method presented in Freville and Plateau [12]. In this method, the constraint coefficients ($\mathbf{A}$) and objective function coefficients ($\mathbf{p}$) are correlated to the constraint coefficient distribution. The constraint coefficients ($\mathbf{A}$) are generated from a uniform distribution with ranges of 1 to 100, 1 to 1000, and 1 to 10000. The correlation of each $\mathbf{a}$ and $\mathbf{p}$ are as follows, where $\mathbf{A}$ (≥ 0) is uniformly distributed over the interval (1, β), and $r_j, \tau \in U(0,1)$.

$$p_j = \left(\sum_{i=1}^{m} A_{ij} \right) / m + 0.5\,\beta\, r_j \qquad \forall j = 1, 2, \ldots, m$$

$$b_i = \tau \sum_{i=1}^{n} A_{ij} \qquad \forall i = 1, 2, \ldots, n$$

Table 4.1 shows the results from this experiment. The problems are all the same size, with 100 variables and 5 constraints. Each problem set is made up of 10 problems, in which the 9 problem sets represent variations in the constraint tightness factor, τ, as defined above, and in the **A** distribution. The difference between the MA-MKP problems and the 1A-MKP problems is the range of **A**. In the 1A-MKP problems, the range of **A** is shown in the first row of the table. For the MA-MKP problems, the range of **A** changes for each of the 5 constraints. The ranges of **A** are set such that the resulting coefficient of variation (CV) of **p** is similar for both the 1A-MKP and MA-MKP problems within a set.

Table 4.1: MKP Comparison — Multiple vs. Single Attributes

1D A dist		**U(1,100)**			**U(1,1000)**			**U(1,10000)**		
τ	**0.25**	**0.5**	**0.75**	**0.25**	**0.5**	**0.75**	**0.25**	**0.5**	**0.75**	
Problem Set	1	2	3	4	5	6	7	8	9	**Avg.**
MA Averages										
CPU Time	15.53	36.26	28.04	575.79	447.05	173.41	241.90	172.03	164.05	206.01
MIP Iterations	184224	239915	188627	7198247	5824636	2758905	3264336	1369109	1223134	2472348
B & B Nodes	91567	116103	94036	3493617	2688241	1260152	1572333	629221	574470	1168860
CV of **p**	7	8	7	45	46	46	474	488	484	178
1A Averages										
CPU Time	5.40	8.35	5.84	4.39	7.79	2.04	10.21	8.40	5.59	6.44
MIP Iterations	72039	57152	39351	61502	116806	30682	141751	122459	38451	75577
B & B Nodes	35734	27795	19730	29789	58035	15148	71958	60492	19419	37567
CV of **p**	5	5	5	47	47	48	501	514	508	187
% MA > 1A										
CPU Time	188%	334%	380%	13011%	5641%	8419%	2269%	1947%	2837%	3892%
MIP Iterations	156%	320%	379%	11604%	4887%	8892%	2203%	1018%	3081%	3171%
B & B Nodes	156%	318%	377%	11628%	4532%	8219%	2085%	940%	2858%	3011%

For each type of problem, Table 4.1 shows the averages over 10 problems for CPU time, the number of MIP Simplex iterations, the number of branch and bound (B & B) nodes visited, and the CV for **p**. The last three rows of Table 4.1 show the average percentage that the

Table 4.2: MKP with Multiple Attributes Comparison — Scaled (MAS) versus No Scaling (MA)

1D A dist	**U(1,100)**			**U(1,1000)**			**U(1,10000)**			
τ	**0.25**	**0.5**	**0.75**	**0.25**	**0.5**	**0.75**	**0.25**	**0.5**	**0.75**	
Problem Set	1	2	3	4	5	6	7	8	9	**Avg.**
MAS Avg										
CPU Time	17.3185	20.0514	13.9778	626.867	507.0962	220.109	291.8927	98.8918	98.0214	210.47
MIP Iterations	203780	272947	189163	7511222	6467788	3493973	3825131	1503139	1439241	2767376
B & B Nodes	100814	133751	94966	3691900	2904949	1607691	1808744	689314	666691	1299869
% MAS>MA										
CPU Time	12%	-45%	-50%	9%	13%	27%	21%	-43%	-40%	-11%
MIP Iterations	11%	14%	0%	4%	11%	27%	17%	10%	-18%	12%
B & B Nodes	10%	15%	1%	6%	8%	28%	15%	10%	16%	11%
% MAS>1A										
CPU Time	221%	140%	139%	14174%	6413%	10714%	2759%	1077%	1655%	4143%
MIP Iterations	183%	378%	381%	12113%	5437%	11288%	2598%	1127%	3643%	4128%
B & B Nodes	182%	381%	381%	12293%	4906%	10513%	2414%	1040%	3333%	3938%

MA-MKP exceeds the 1A-MKP in each of the categories. It is clear that these experiments support the conjecture that the MA-MKP is more difficult to solve than the 1A-MKP.

A second round of testing was performed on these same problems in which the constraints are scaled. The coefficients in each constraint are divided by the constraint's right-hand side value so that the scaled right-hand side is equal to one and the scaled constraint coefficients are between zero and one. Each A_{ij} then represents the percentage of that knapsack that the variable would consume if selected. Table 4.2 presents the results of this scaled comparison.

Table 4.2 shows that scaling does not make the problems easier for CPLEX to solve (this may be caused by the fact that CPLEX includes some scaling procedures that are probably used even on our "unscaled problems"). For the scaled problems, the CPU time, number of MIP iterations, and the number of branch and bound nodes visited remains close to the same as when the problem is not scaled. As shown in the final three rows of Table 4.2, on average, the MA-MKP still requires significantly more CPU time, MIP iterations, and branch and bound nodes than is required for the 1A-MKP. Therefore, these results support the conjecture that the MA-MKP is more difficult to solve than the 1A-MKP, and that scaling does not alleviate this difficulty.

In summary, although the MA-MKP and 1A-MKP are mathematically equivalent, the experimentation indicates that the MA-MKP is more difficult to solve than the 1A-MKP. The next step is to identify which, if any, MIP solution methods work better than others for problems with the MA-MKP structure.

4.2 Multiple-Attribute Experimentation

This section measures the impact of solution techniques on the solution time of the sourcing problem. As previously shown, the sourcing problem has characteristics that make it more difficult to solve than the standard MKP addressed in the literature. Solution techniques that are effective for the standard MKP may not be useful for the sourcing problem. Therefore, it is necessary to evaluate various techniques and their effectiveness on the sourcing problem.

Three sets of problems are used for calculating the results in this section: single-plant insourcing only (as described in Chapter 3) with 100 parts and 5 constraints; single-plant sourcing (insourcing and outsourcing) with 200 parts and 5 constraints; and single-plant sourcing with 298 parts and 5 constraints. As in previous result presentations, in all of the tables in this section, each entry is an average of 10 problems, and each table covers 90 problems.

4.2.1 Insourcing versus Sourcing

The experimentation in this section uses problems in the form of both the single-plant insourcing problem and the single-plant sourcing problem (SPSP) as defined in Section 3.3.3. The model of the SPSP includes both insourcing and outsourcing, and is reviewed here with

$x_i = 1$ when part i is insourced and $y'_h = 1$ when currently loaded part h remains in-house.

$$\begin{array}{lll}
\text{Maximize} & \sum\limits_{i=1}^{n} p_i x_i - \sum\limits_{h=1}^{k} c_h \left(1 - y'_h\right) & \\
\text{subject to} & \sum\limits_{i=1}^{n} t_{ij}^{m} x_i + \sum\limits_{h=1}^{k} t_{hj}^{m} y'_h \leq b_j^{m} + \sum\limits_{h=1}^{k} t_{hj}^{m} & \forall j = 1, 2, \ldots, m \\
& \sum\limits_{i=1}^{n} t_{ij}^{\ell} x_i + \sum\limits_{h=1}^{k} t_{hj}^{\ell} y'_h \leq b_j^{\ell} + \sum\limits_{h=1}^{k} t_{hj}^{\ell} & \forall j = 1, 2, \ldots, m \\
& x_i \in \{0, 1\} & \forall i = 1, 2, \ldots, n \\
& y'_h \in \{0, 1\} & \forall h = 1, 2, \ldots, k
\end{array}$$

The first issue to address is if the addition of outsourcing adds difficulty to the model. To examine this, a comparison is made between a strictly insourcing problem set and an SPSP set that includes outsourcing. The coefficients for the insourced parts (x_i) in the SPSP are the exact same coefficients as in the insourcing problems. For the outsourced parts, (y_h), the coefficients are randomly generated using the same method and parameters as for the insourcing coefficients. Effectively, both the insourcing problem and the SPSP are of the MA-MKP type, with the only difference being that the SPSP has twice as many parts. In the experimental problems, the insourcing problems have 100 parts and 5 constraints, while the SPSP has 200 parts and 5 constraints. These two problem sets are both solved to optimality using CPLEX under the default settings, and the results from these experiments are shown in Table 4.3. The table entries represent the average CPU time (of ten problems) required to solve the problems optimally.

Comparing the CPU time for the two types of problems, on average, the SPSP requires 1.5 times the CPU time, but it is also twice as large. In some cases, for problem sets 3, 8, and 9, the CPU time is actually less for the sourcing problem. Therefore, we can conclude that the addition of outsourced parts, although increasing the size of the problem, does not imply a correlated increase in the time required to solve the problem. Hence, both types of problems will be used to test the various solution techniques in this section.

Table 4.3: Insourcing Problem CPU Solution Times

1D A dist	U(1,100)			U(1,1000)			U(1,10000)			
Tau	0.25	0.5	0.75	0.25	0.5	0.75	0.25	0.5	0.75	
Problem Set	1	2	3	4	5	6	7	8	9	**Average**
Insourcing	15.5	36.3	28.0	575.8	447.1	173.4	241.9	172.0	164.1	206.01
SPSP	33.8	86.7	25.0	788.0	857.9	355.3	296.8	144.9	115.3	300.39

4.2.2 Parameter Control

In this section the specific problem parameters are addressed with respect to their impact on solution time and sensitivity. First, the problems are scaled so that all the constraint coefficients are on the same scale. Next, the problem parameters are rounded to test the effects of data accuracy.

Scaling

In Section 4.1 it is demonstrated that the MA-MKP is more difficult to solve than the 1A-MKP. This continues to hold true when the problems are scaled so that the coefficients are between zero and one. However, this comparison between the scaled and unscaled problems may be inaccurate because included in the default CPLEX settings is a coefficient preprocessing function. It is possible that a coefficient reduction that is similar to scaling in the previous experiment is completed during the default preprocessing function. Therefore, an experiment is conducted with the scaled MKP problems and the default coefficient preprocessing disabled. The results are presented in the No Preprocessing section of Table 4.4 (along with the results from the original scaled versus unscaled experiments). With preprocessing disabled, there is still no significant difference between the scaled and unscaled solution times.

In both scenarios, with and without preprocessing, the scaled problems on average require slightly more CPU time, but the difference is very small and only on average, not across all the problems. Therefore, the MA-MKP remains difficult to solve even when scaled to look

Table 4.4: Scaled and Unscaled CPU Solution Times

Problem Set	1	2	3	4	5	6	7	8	9	**Average**
CPLEX Default										
Unscaled MA-MKP	15.5	36.3	28.0	575.8	447.1	173.4	241.9	172.0	164.1	206.0
Scaled MA-MKP	17.3	20.1	14.0	626.9	507.1	220.1	291.9	98.9	98.0	210.5
1A-MKP	5.4	8.4	5.8	4.4	7.8	2.0	10.2	8.4	5.6	6.4
No Preprocessing										
Unscaled	13.4	16.7	13.4	609.8	529.4	174.0	232.7	97.1	73.5	195.6
Scaled	13.9	18.2	12.5	596.8	484.9	216.0	349.9	96.2	93.1	209.1

like a 1A-MKP.

The scaling process may require coefficients to be truncated or rounded. Since on average, the scaled problems are slightly more difficult to solve, this brings rise to the next consideration of whether rounding and truncation adds difficulty to the problem.

Rounding

The rounding that occurs from scaling is small enough that the solution obtained is the same as the solution from the original problem. This section looks at more significant rounding to test the impact on the solution procedure. The experimentation addresses two issues: first, does rounding increase the difficulty in solving the problem, as mentioned in the previous section? And second, how sensitive is the solution procedure and the solution accuracy to slight changes in the problem parameters? This latter issue would arise when the problem parameters are estimates, which is likely to occur when using industry data.

To test these issues, two types of problems are compared. The insourcing problem set, solved with CPLEX under the default settings, is again used as the baseline for comparison. These same problems are then rounded, to the nearest ten, to form the second problem set that is also solved with CPLEX under the default settings.

The results are shown in Table 4.5. The first row, labeled % CPU Reduction, is the percent

Table 4.5: Rounded Solutions

Problem Set	1	2	3	4	5	6	7	8	9	**Average**
% CPU Reduction	93.3	98.8	97.3	27.7	-14.7	38.8	-11.5	35.7	51.9	46.4%
# Vars same	89.1	89.9	88.7	95.5	91.6	91.2	100.0	98.0	100.0	93.8
Soln Gap (%rd>reg)	0.18	-0.03	0.12	0.01	0.00	0.00	0.00	0.00	0.00	0.04%

reduction in CPU time when solving the rounded problem versus solving the regular problem with default CPLEX settings. On average over the nine problem sets, solving the rounded problem reduces the run time by 46.4%. Therefore, the initial conjecture that the problem becomes more difficult with rounding is shown to be false. However, this makes sense because rounding is like replacing constraints with Chvatal-Gomory cuts of them, which can be expected to give a tighter representation.

The reduction in time to solve a rounded problem leads to the possibility that a problem can be estimated and solved more quickly. However, before doing this, it would be helpful to know how sensitive the final solution is to changes in the parameters. Therefore, the solution from the rounded problem is compared to the original solution and the results are summarized in the remainder of Table 4.5. The second row shows the average number of parts (out of 100 total parts) that are assigned to the same value in both the regular problem and the rounded problem solutions. As the range on the coefficients increases, more parts are assigned the same value. This makes sense since the rounding to the nearest ten affects the larger coefficients less than the smaller coefficients. However, even when up to 11% of the parts are assigned different values, the difference between the objective function values is quite small. The final row of Table 4.5 shows the percent change in the solution value from the regular problem solution to the rounded problem solution. (A negative value, as in Problem Set 2, can occur because when the coefficients are rounded, both the capacity of the knapsacks and the amount of capacity a part consumes changes, and solutions that are infeasible to the original problem may now be feasible to the rounded problem.) Therefore, the rounded solution is a good estimate of the original solution, particularly when the range of the coefficients increases.

4.2.3 Algorithm Control

This section addresses control of the branch-and-bound algorithm. Various strategies are employed, each dealing with the order in which the branch-and-bound tree is developed. First, traditional depth- and breadth-first search methods are compared. Next, the order in which the variables are branched on is prioritized based on a ranking. A few different ranking strategies are tested.

Depth versus Breadth

In the previous section on rounding, multiple solutions were found that had similar values. If a good solution can be found quickly, it can help to eliminate poor solutions quickly. In light of this, it is worth investigating the order in which solutions are generated and evaluated in the solution procedure.

In this section, control of the branch-and-bound algorithm is evaluated with respect to branching order, or how the tree is developed. In a depth-first search, at each node, if possible, the next node considered is a child of the current node. It is likely that feasible solutions will be found deep in the tree; therefore, this method is likely to find a feasible solution quickly. Another method is a breadth-first search in which all the nodes at one level of the tree are evaluated before the children of that level are considered.

On the same set of multiple-attribute problems as in the previous experiments, depth-first and breadth-first searches are compared. This is done by solving the problems using both a pure depth-first search and a pure breadth-first search. Table 4.6 shows the results from the two techniques compared with results from default CPLEX, which uses a combination of depth- and breadth-first. The values in the table are the percent reduction in CPU solution time when using the pure depth- or breadth-first compared to the default CPLEX.

Although, the breadth-first search is on average not as efficient as the default CPLEX, it is significantly faster than depth-first. Additionally, breadth-first is significantly better than

Table 4.6: % Reduction in CPU Solution Times with Depth- and Breadth-First

Problem Set	1	2	3	4	5	6	7	8	9	**Average**
Depth-first	-273	-49	18	-199	-304	-215	-110	-184	-35	-150%
Breadth-first	5	47	46	-79	-52	-4	-61	53	49	0.3%

the default CPLEX for 5 of the 9 problem sets and for 66 of the 90 problems. The poor performance of the depth-first search indicates that one difficulty in solving MA-MKP may be in quickly eliminating poor solutions. This is consistent with the rounding results section where multiple solutions provide similar objective function values. When multiple good solutions exist, partial solutions appear good until deep in the tree.

Branching Priorities

As in the previous section, the control of the branch-and-bound algorithm is addressed here. A list is created that sets the priority of the parts with respect to the branching order. That is, of the possible parts that can be branched on, the part with the highest priority is selected.

Three different methods of ranking the parts are defined and tested. All three methods are based on maximizing the composite profit to cost ratio, where a part's profit and cost are respectively p_i and t_{ij} for insourced parts, and c_h and t_{hj} for outsourced parts. For simplicity in defining the rankings, the outsourcing profits will also be represented as p_i, and costs as t_{ij}, where i is indexed from 1 to $n+k$, representing both the insourced and outsourced parts.

Ranking of parts by the maximum profit to cost ratio is simple with only one constraint. In that scenario, each part has one profit value and only one cost value. However, with more than one constraint, the composite cost factor is not easily defined, since for each part, each constraint has a different cost.

The three rankings differ by how the composite cost factor, V_i (the denominator in the profit to cost ratio), is calculated. The first ranking defines V_i as the sum of the costs across all constraints. The second ranking uses only the cost of the constraint with the most expensive coefficient, effectively selecting the best worst case. The third ranking is based on the heuristic to solve the MKP by LouLou and Michaelides [25].

In the third ranking, parts are selected one at a time, based on a criterion, and added into the knapsack. The order in which they are selected is then used as the priority ranking with the first selected being on the top of the list. This is different from the above rankings in that once a part is selected, the criterion changes and is recalculated for the remaining parts. A few items need to be defined to understand the selection criterion. DA_j is the percent of capacity consumed on machine j thus far. As parts are selected this value increases. SC is the list of remaining candidate parts, and the percent of each knapsack a part consumes is $a_{ij} = t_{ij}/b_j$.

For all the parts in SC, the heuristic calculates the criterion, and the part with the largest value is selected and removed from the set SC. Like the previous rankings the criterion is a profit to cost ratio; however, it is different in that the cost factor, V_i, changes after each selection and it is made up of three factors. The first factor, $DA_j + a_{ij}$, is the total consumption of machine j capacity by all the parts selected so far plus part i. When this consumption of a machine is high, this factor increases V_i, ultimately decreasing the profit to cost ratio for part i in this iteration. The second factor, $\sum_{k \epsilon SC} a_{kj} - a_{ij}$, is the potential future demand on machine j (from the remaining parts in SC) after part i is selected. When this projected future demand is large, V_i is large to lower the priority of parts that use of this machine. The third factor, $1 - DA_i - a_{ij}$, is the remaining capacity on machine j after part i is selected. If the machine is close to capacity, this quantity is small and V_i is large to discourage use of the machine.

Ranking Criteria

$\arg\max_i \left\{ \dfrac{p_i}{V_i} \right\}$, where

1. $V_i^1 = t_{ij}$

2. $V_i^2 = \max_j \{t_{ij}\}$

3. $V_i^3 = \max_j \left\{ \dfrac{(DA_j + a_{ij}) \left(\sum_{k \epsilon SC} a_{kj} - a_{ij} \right)}{(1 - DA_i - a_{ij})} \right\}$

To compare the value of these branching priorities versus each other and CPLEX default branching, problems are solved in CPLEX using each of the three rankings as inputs. For each problem, each ranking strategy has a corresponding list that controls the order in which parts are branched on. For these experiments, the problem sets used are the same as those in Section 4.2.1 for the SPSP with 100 insourced parts, 100 outsourced parts, and 5 constraints. Table 4.7 shows the results from the three ranking strategies used as branching priorities. The first rule shows some benefit for the problems with the smaller coefficient range (sets 1-3), but otherwise, default CPLEX significantly outperforms CPLEX with the above branching priorities. Although the priorities are not effective in this scenario, these ranking strategies will be used with some success in Chapter 5 as part of a heuristic approach.

Table 4.7: % Reduction in CPU Solution Time with Branching Priorities

Problem Set	1	2	3	4	5	6	7	8	9	**Average**
Rank Rule 1	64	55	29	-456	-278	-397	-529	-654	-367	- 281%
Rank Rule 2	-24	-50	-61	-796	-187	-498	-375	-1218	-648	-429%
L&M rank	-69	-93	-73	-706	-384	-892	-961	-703	-413	-477%

4.2.4 Solution Space Control

This section deals with approaches that attempt to eliminate solutions of either the entire problem or for subproblems. These approaches includes a Lagrangean bound, covers, and cuts.

Table 4.8: % Reductions in CPU Solution Time with a Lagrangean Bound

Problem Set	1	2	3	4	5	6	7	8	9	**Average**
Root Node Only	14	46	52	-2	-1	-2	0	41	53	22%
Every Node	-50477	-24243	-144176	-14604	-19602					-50620%

Lagrangean Bound

A Lagrangean bound is implemented in two ways. First, the bound is generated for just the root node. This bound is input into CPLEX with the problem and solved using the default settings. Under the second method, the algorithm is stopped at each node, a bound is generated for the subproblem, and submitted back to the subproblem to continue until the problem is solved to optimality. The Lagrangean bound is calculated using the subgradient optimization method, and the multipliers and step size are determined as in Gavish and Pirkul [15].

To calculate the Lagrangean bound, one constraint remains active, while the others are added to the objective function and weighted by multipliers. The remaining 0-1 knapsack problem (single-dimensional) is solved to optimality. This procedure is repeated with each of the constraints remaining active, and the best solution is used as the bound.

The results from testing these two bounding methods are shown in Table 4.8. The values in the table are the average percent reduction in CPU time, when the Lagrangean bound is used instead of the default CPLEX, to solve the problem to optimality. On average, calculating the Lagrangean bound at the root node decreases the solution time by 22%. However, solving the bound at every node clearly takes much longer. This bound is expensive (with respect to time) because a knapsack problem is solved for each constraint.

A possible alternative is to solve the knapsack problems partially instead of to optimality, reducing the time required to calculate the bound. For each node, instead of solving the relaxed problem optimally a greedy heuristic is used. Since the relaxed problem has only one constraint, the variables are ranked by the benefit to cost ratio. Following the ranking order, variables are added when capacity is available. However, this experiment very little increase in run time is observed over solving the relaxed problem. When analyzed further, the long run time can be attributed to the stopping and starting of CPLEX at every node. However, the moderate success of the root node Lagrangean bound indicates that further bounding methods could potentially reduce the CPU time further. In the next section, additional constraints and cuts are generated in an effort to reduce the solution space.

Extended Cover

In this section an extended cover inequality is generated and added to the root problem. With the extra constraint, the problem is input into CPLEX to again compare the CPU times to examine the effect of the more tightly constrained problem.

The solution procedure, developed by Gabrel and Minoux [13], generates the most violated extended cover inequalities with an exact solution approach to solve the separation problem. The solution to the separation problem defines a minimal-dependent set used to generate the extended cover inequality.

Gabrel and Minoux [13] show that the use of extended cover inequalities is in general more effective than the default CPLEX cover inequality generation. However, the problems used in testing, as in other MKP literature, are single-attribute problems in which all the constraint coefficients are generated from the same distribution. This extended cover method was tested on the multiple-attribute problems for the insourcing problem with 100 parts and 5 constraints. The results in Table 4.9 show that there is a moderate improvement (on average 17%) in the CPU time required to solve the problem to optimality when the cover is included. It is most effective on the problems with both small and large coefficient ranges, but is worse in the middle range. The time reduction reported in the table includes only the time to solve the MKP with the additional constraints. The time to generate the ECI is not included. However, as with the Lagrangean bound, this procedure requires solving multiple 0-1 knapsack problems and it is expensive with respect to the time required to generate these

Table 4.9: % Reduction in CPU Solution Time with Extended Cover Inequalities

Problem Set	1	2	3	4	5	6	7	8	9	**Average**
% Reduction w/ECI	-2	46	48	-15	4	-13	7	41	49	17%

inequalities.

Cuts

The final set of experiments that test methods constraining the problem are executed using the various cut options for 0-1 integer programs available in CPLEX: GUB, Gomory, Cover, and Disjunctive. The tests are executed on the SPSP with two problem sizes, 200 parts with 5 constraints, and 298 parts with 5 constraints. The results from these experiments are shown in Table 4.10.

For a set of binary variables, the GUB (generalized upper bound) constraints take a form such that the sum of the variables is less than or equal to one. This is based on the idea of splitting the feasible region into two sections instead of branching on an individual part. However, with respect to CPU time, using only the GUB cut option is dominated by the default CPLEX cuts for both problem sizes.

Gomory cuts are generated by applying integer rounding to a basic variable row in the optimal linear programming (LP) tableau in which the variable is fractional. On average, with only the Gomory cut option selected, slightly less CPU time is required than with the CPLEX default.

The cover cut option generates minimal cover inequalities. This is a similar cut to the extended cover inequalities in Section 4.2.4; however, with this CPLEX option there is no guarantee that the generated constraint yields a facet of the convex hull. This cut option performs well as compared to the default CPLEX and is comparable to the Gomory cut option.

Table 4.10: % Reduction in CPU Solution Time with Cuts

Problem Set	1	2	3	4	5	6	7	8	9	**Average**
100-100-5										
GUB cuts	-146	-150	-148	-119	-108	-115	-150	-173	-172	-142%
Gomory Cuts	-36	-4	-49	5	12	27	19	7	9	-1%
Cover Cuts	7	9	7	15	20	21	7	0	1	10%
Disjunctive Cuts	26	0	9	10	25	13	17	-65	-162	-14%
149-149-5										
GUB cuts	-129	-213	-150	-250	-222	-278	-179	-229	-218	-208%
Gomory Cuts	4	-7	22	5	5	13	-6	2	20	7%
Cover Cuts	7	4	5	6	7	6	2	-1	-2	4%
Disjunctive Cuts	-136	-214	-151	-246	-212	-271	-171	-225	-214	-204%

Finally, the disjunctive cut option uses the knowledge that each variable is either less than or equal to zero or is greater than or equal to one. Disjunctive cuts are generated on the subproblems that are valid for the LP feasible region, but not for the root problem. This method performs poorly compared to the the other cuts and the default CPLEX options.

In summary, of the various cut options, Gomory and cover cuts perform similarly, and with respect to CPU time, perform on average better than the default CPLEX and the other cuts. For each problem set, either the Gomory or cover option requires less CPU time than the default CPLEX. Generating the Gomory cuts involves rounding; therefore, given earlier results with rounding, it is not surprising that the Gomory cuts perform well.

4.2.5 Summary

As shown in the Chapter 4, with respect to CPU solution time the multiple attribute multi-dimensional knapsack problems (MA-MKP) are more difficult to solve than the single dimensional problems. Since the sourcing problems are modeled as an MA-MKP, we are interested in identifying solution techniques that work well on this type of problem.

This chapter summarizes the results from experimentation on both preprocessing and algorithmic control techniques to solve the MA-MKP. Some of the techniques are more effective than others and provide some insight into the difficulty in solving MA-MKPs.

First, when the coefficients are rounded, the problem on average requires 46% less CPU time than solving the original problem with default CPLEX. In addition, the solution obtained from the rounded problem is on average within 0.04% of the optimal solution. As the coefficient range increases, the solutions are identical in most cases. This combined with the dominance of a breadth-first search over a depth-first search indicates that a potential difficulty in solving the MA-MKP is that since multiple good solutions can exist (shown by the close solutions found when rounding) it may be difficult to eliminate partial solutions until deep in the tree.

Second, the Lagrangean relaxation on average reduced the required CPU time by 22%, even though a bound was generated for only the root node. It is expensive to generate the bound because a regular knapsack problem is solved for each constraint. Therefore, to generate a bound for every node, the solution to the Lagrangean problem needs to be estimated. However, given the success of the root node bound, it is likely to reduce even further the CPU time required to solve the multiple-attribute problems to optimality.

Finally, the cover and Gomory cuts outperformed the CPLEX default settings as well as the other cuts. Although, these methods made significant reductions in the solution time, none reduced the solution time to the scale of the single-attribute problems. Even with the most effective techniques, the multiple-attribute problem still requires about 25 times the CPU time to find the optimal solution. For this reason (in addition to the size of industry problems) heuristics will be discussed in the next chapter.

Chapter 5

Heuristic

In the previously discussed techniques, an improvement in the solution time is seen with a few of the MIP exact approach methods, yet all are expensive with respect to CPU time. Since industry problems are considerably larger than the experimental problems, it is worth exploring heuristic solution techniques. The linear programming (LP) relaxation of the problem and simple heuristics that start with the LP-relaxed solution are addressed in this chapter.

5.1 LP Relaxation

Before using the LP relaxation, it is important to know how the LP solution compares to the optimal solution. The results from this comparison are shown in Table 5.1 for the problems with 100 insourced parts and 100 outsourced parts. The first two rows clearly show, as expected, that the CPU time required to solve the LP is negligible compared to the time required to solve the IP to optimality. The next row is the average number of fractional variables (related to part selection) in the LP solutions. Soyster, Lev, and Slivka [36] show that for multi-dimensional knapsack problems at most m part selection variables will be

Table 5.1: Single Plant Sourcing Problem LP Relaxation

Problem Set	1	2	3	4	5	6	7	8	9	**Average**
CPU-CPLEX default	34	87	25	788	858	355	297	145	115	300
CPU-LP relaxation	0.06	0.03	0.03	0.06	0.03	0.04	0.04	0.03	0.05	0.04
# Fractional in LP	4.9	5.0	5.0	5.0	5.0	5.0	5.0	5.0	4.9	5.0
% Gap										
LP relax - IP	0.41	0.28	0.18	0.24	0.12	0.08	0.29	0.12	0.07	0.20%
IP - LP rd down	8.86	4.25	3.29	11.47	4.24	3.16	11.22	4.88	3.65	6.11%

fractional, where m is the number of constraints. Since the test problems have 5 constraints, all of the problems have a maximum of 5 fractional parts in the LP relaxation solution.

The next two rows display the percent gap between solutions. The row labeled LP relax - IP, is the gap between the objective function value of the IP optimal solution and the LP (fractional) solution. The IP - LP rd down row is the gap between the objective function value of the IP optimal solution and the LP feasible solution (fractional parts rounded down). Note that the gap is still small when comparing the IP to the LP feasible solution (6.11%); however, there is still room for improvement.

5.2 LP-Based Heuristic

Discussed next is a simple heuristic that capitalizes on the knowledge that the number of fractional parts in the LP solution is at most equal to the number of knapsack constraints and the assumed small gap between the feasible LP solution and the IP solution. The idea behind the heuristic is to start with the LP feasible solution determined from rounding down the fractional parts in the LP solution, then if possible, add any extra parts into the knapsack (changing the variables currently set to zero to one). However, this requires a method to evaluate which parts should be added to the knapsacks.

Assume there exists a feasible solution and a list of candidate parts (variables currently set to zero) to be added to the knapsacks. A simple greedy method is to rank the candidate parts, based on some criterion, and then add as many as possible, in order of the ranking, while maintaining feasibility.

Four different rules are used to rank the candidate parts. Each one is based on the idea of ranking the parts by a composite benefit to cost ratio, or the "bang for buck." However, as discussed before in the branching priorities section, Section 4.2.3, the cost factor is difficult to calculate because of the multiple constraints. Therefore, each rule has a slightly different approach to calculating the cost factor. The first two ranking rules are the same as rules 1 and 2 in the branching priorities section. The first rule takes the cost factor to be the sum of a part's coefficients across the constraints. The second and third rules are similar to each other; however, where the second rule creates a worst case ratio, the third rule generates the best case ratio.

The fourth rule is also based on a composite benefit to cost ratio. In this case, the structure is similar to rule 1, but with surrogate multipliers as weights on the coefficients. These multipliers come from the strongest surrogate constraint, defined as $(\mu^*)^t Ax \leq (\mu^*)^t b$, such that $s(\mu^*) = \min_{\mu} \{S(\mu)\}$, where $S(\mu) = \max\{cx \mid \mu(Ax - b) \leq 0, x \in \{0,1\}\}$, and μ is a positive vector of size m. To get the strongest surrogate constraint, solve for the set of multipliers, μ, as defined in problem $S(\mu)$. Of these solutions, the multipliers that generate the minimum solution, μ^*, are used to generate the strongest surrogate constraint. The problem $S(\mu)$ is known as the surrogate relaxation. In this relaxation, the original constraints are replaced by the single strongest surrogate constraint. This effectively creates a 0-1 knapsack problem, and the benefit to cost ratio can be calculated as in the 0-1 knapsack problem. Therefore, for the surrogate ranking rule, the coefficients in the surrogate relaxation, $\sum_{j=1}^{n+k} \mu_j t_{ij}$ are used as the cost factors.

This surrogate ranking procedure is similar to that developed by Pirkul [33]. The difference is that Pirkul started with all variables set to zero and used the surrogate ranking to determine the order in which the variables would be added. This procedure starts with the feasible LP solution and ranks only the remaining variables as possibilities to add into the knapsacks.

Surrogate relaxations are often overlooked as an effective bounding procedure because the feasible region for optimal multipliers is non-convex [26]. In this heuristic, the method of Gavish and Pirkul [15] is used to find the multipliers. They take the optimal dual variables of the LP problem as the surrogate multipliers, and show that the bound produced by the surrogate relaxation problem (using the LP dual variables as multipliers) is at least as good as the LP bound. Therefore, in this heuristic, the LP dual multipliers are used as the surrogate multipliers.

Ranking Rules

1. $R_i^1 = \max_i \left\{ \frac{p_i}{\sum_{j=1}^{n+k} t_{ij}} \right\}$

2. $R_i^2 = \max_i \left\{ \frac{p_i}{\max_j \{t_{ij}\}} \right\}$

3. $R_i^3 = \max_i \left\{ \frac{p_i}{\min_j \{t_{ij}\}} \right\}$

4. $R_i^4 = \max_i \left\{ \frac{p_i}{\sum_{j=1}^{n+k} \mu_j t_{ij}} \right\}$

In each of the k rules, $k = 1, 2, 3, 4$, the candidate variables are ranked on R_i^k. The variable with the largest R_i^k is checked first to see if it is feasible to add that variable to the knapsacks (change the value from zero to one).

Table 5.2 displays the results from using each rule, as well as the best of the four, to rank the candidate variables. Although rank rule 1 appears to be the best on average, no one rule comes close to dominating for all problem types. The top of the table is for problems with 100 outsourced parts, 100 insourced parts, and 5 constraints. The bottom section shows the results for larger problems with 149 outsourced parts, 149 insourced parts, and 5 constraints.

The first column of Table 5.2 is the same calculation as the row in Table 5.1 labeled IP -

LP rd down. This is the gap between the rounded down feasible LP solution and the optimal IP solution and will be used as the baseline reference for comparing the ranking rules. Notice that as the number of parts increases from 200 to 298, this gap decreases from 6.11% to 3.65%. Under each ranking rule, the two columns represent the number of parts changed from zero to one, and then the size of the gap after the extra parts are added. The min gap column is the average gap when the best rule is selected for each problem. Each row represents 10 problems, so the value shown in the table is the average over the 10 problems when the best rule is selected for each problem. This min gap will always be less than or equal to any one rule. If it is equal to any one rule, then that rule dominates the others for all 10 problems. However, no one rule dominates the others over all the problems, although they all perform quite well. Comparing the results based on problem size, as the problem gets larger, the gap decreases. Since the test problems are small compared to actual industry problems, the results indicate that these simple greedy heuristics are very promising for larger problems.

Although for the small test problems this heuristic performs well, these solutions can also be used for solving the problems to optimality. The next set of results is from an experiment using the solution from each ranking rule as an initial bound to solve the problem to optimality. The results are shown in Table 5.3 for the problems with 149 outsourced parts, 149 insourced parts, and 5 constraints. Compared to the default CPLEX settings, using the heuristic solution as an initial solution decreased the CPU time required to solve the problem to optimality. On average over the 90 problems, using the best heuristic solution as the initial bound, the ranking rules decrease the CPU solution time by 6.5%.

Since no one rule performs the best in all situations, how do we compare and select the appropriate rule? All four ranking rules perform well with respect to generating a solution close to the optimal, and no one rule dominates the others. However, they are fast enough that all four rules can be run to generate very good solutions, such as those found in the far right column of Table 5.2, labeled min avg. Using the heuristics solutions as initial bounds to solving the problem optimally produces a larger discrepancy between the rules. On average across each problem set, rank rule 3 and the surrogate ranking dominate rules 1 and 2.

Table 5.2: Comparison of LP Relaxation + Extra Parts

Prob. Set	% Gap	Rank Rule 1 # added	Rank Rule 1 % Gap	Rank Rule 2 # added	Rank Rule 2 % Gap	Rank Rule 3 # added	Rank Rule 3 % Gap	Surrogate # added	Surrogate % Gap	Min Gap
100-100-5										
1	8.86	2.1	3.32	2.3	4.00	2.2	2.74	2.3	3.48	2.45
2	4.25	1.8	1.40	1.9	1.98	1.5	1.53	1.9	1.37	0.98
3	3.29	2.2	0.89	2.7	0.98	2.2	0.90	2.1	1.14	0.66
4	11.47	2.1	3.82	2.4	2.87	2.2	4.01	1.8	4.51	2.22
5	4.24	1.5	1.44	1.5	1.49	1.6	1.39	1.4	1.59	1.05
6	3.16	1.7	1.01	1.8	0.90	1.9	0.99	1.6	1.12	0.68
7	11.22	2.2	3.27	2.1	3.51	2.0	4.23	2.1	3.01	2.39
8	4.88	1.8	1.49	1.8	1.62	1.7	1.91	1.6	1.89	1.45
9	3.65	2.2	0.79	2.1	1.02	2.2	1.03	1.9	1.28	0.65
Avg.	**6.11%**	2.0	**1.93%**	2.1	**2.04%**	1.9	**2.08%**	1.9	**2.16%**	**1.39%**
149-149-5										
1	5.38	2.3	1.26	2.4	1.41	2.0	2.00	2.3	1.20	1.03
2	2.98	2.1	0.77	2.3	0.81	2.2	0.83	2.2	0.70	0.59
3	1.83	1.9	0.46	1.8	0.64	1.6	0.71	1.8	0.56	0.39
4	6.68	2.3	1.18	2.4	1.06	2.2	1.76	2.2	1.50	0.92
5	3.28	1.9	0.90	1.9	0.88	1.7	1.30	1.9	0.94	0.75
6	2.75	2.6	0.56	2.7	0.45	2.3	0.86	2.4	0.72	0.35
7	5.21	1.6	1.49	1.5	1.78	1.5	2.10	1.6	1.40	1.29
8	3.04	1.7	0.94	2.0	0.70	1.9	0.92	1.9	0.69	0.47
9	1.72	1.5	0.51	1.5	0.50	1.5	0.56	1.5	0.53	0.43
Avg.	**3.65%**	2.0	**0.90%**	2.1	**0.91%**	1.9	**1.22%**	1.98	**0.92%**	**0.69%**

Table 5.3: % Reduction in CPU Solution Time with Heuristic Solution as Initial Solution

	298 parts - 5 cons.				
Prob. Set	Best as Init Solu	Rank Rule 1	Rank Rule 2	Rank Rule 3	Surrogate Rule
1	5.31	3.80	-1.94	9.07	8.17
2	4.99	3.93	-3.00	5.57	5.62
3	5.73	5.72	-2.08	7.59	7.24
4	6.89	6.24	3.10	7.09	6.51
5	6.21	5.71	1.93	7.81	6.27
6	7.25	6.17	5.91	7.59	7.82
7	7.08	-10.48	6.25	9.58	8.91
8	7.69	-5.87	5.87	8.49	7.34
9	7.52	-8.04	5.55	7.02	6.20
Average	**6.52%**	**0.80%**	**2.40%**	**7.76%**	**7.12%**

Table 5.4: Rule Effectiveness

	% Best Value	**% Min CPU**	**% Effective**
Rule 1	30%	16%	7%
Rule 2	24%	6%	23%
Rule 3	12%	46%	36%
Surrogate	33%	33%	30%

An important point to notice is that the best heuristic solution does not imply that using it as a bound will generate the optimal solution in the minimal time. Table 5.4 addresses this issue by comparing the different rules based on their performance with respect to the objective function value and the optimal CPU time (when the heuristic result is used as an initial bound when solving for the optimal solution). The first column, % Best Value, shows the percent of problems in which each rule generates the best solution. Here, the surrogate ranking shows the best results. However, as previously mentioned, the heuristic is fast enough that all four rules can be run to be ensured of the best heuristic solution on every problem. The second column in Table 5.4, % Min CPU, shows the percent of problems for which each rule's solution as a bound produces the minimum CPU time for the optimal solution. Here, rule 3 performs the best, despite its poorer heuristic solution values. The final column, % effective, is a combination of the previous two. It represents a rule's effectiveness as the percent of problems in which a rule produces the best solution value, and that value as a bound also produces the minimum CPU time to solve for the optimal solution. From this, rule 3 is the most "effective" rule, with the surrogate also performing well. This is interesting because rule 3 is a much simpler rule to calculate than the surrogate rule.

5.3 Summary

Solving the MA-MKP to optimality is expensive with respect to CPU time; therefore, it is logical to venture into the area of heuristics. The simple heuristic discussed in Section 5.2 starts with the rounded (feasible) LP solution, then, parts currently not included are added into the knapsacks one at a time based on a ranking rule. Although the solution is not necessarily optimal, it is very close, and as the size of the problem increases, the gap with respect to the optimal solution decreases. An additional benefit of this heuristic solution is to use it as a bound. Using the heuristic to generate an initial solution yields a 6.5% reduction in CPU solution time. The primary use of the heuristic is the ability to quickly obtain very good solutions that improve as the size of the problem increases—which is when a heuristic is needed the most.

Chapter 6

Industry Data Sets

The focus of the previous chapters is on the structure of the problem, in particular the classification of multiple- or single-attribute. The multiple-attribute problems are of interest primarily because industry problems fall into this category. However, the experimental data sets used in Section 4.2 that possess the multiple-attribute structure are generated using the procedure suggested by Freville and Plateau [12], and in general do not take into account an industry point of view. Similarly, in current literature, standard test problems, also generated by methods such as the Freville and Plateau procedure [12], are used to compare the efficiency of solution approaches. However, the structure of industry data and the capability of algorithms to solve these data sets varies greatly from the standard test problems.

Since this research is industry motivated, it is important to consider the characteristics of industry data when generating data sets for testing. In this chapter data sets are generated (and tested) that have the multiple-attribute structure; but, additionally, the actual values and interactions between the data reflect industry data. We identify characteristics of industry data that differ from standard literature data and illustrate the impact of these characteristics on the performance of MIP methods to solve the MKP.

6.1 Industry Data Versus Literature Data

Although standard data sets are useful and necessary for comparing solution procedures, the data notably affect the solution time. This affect is evident in the results presented in Chapter 4 that demonstrate the increase in solution time when changes are made to the constraint coefficients forming the multiple-attribute problem. Industry data sets can be very different than the standard sets, or those generated using methods from the literature. Using the data collected from the sheet metal facilities (discussed in Chapters 1 and 3) as motivation, this section characterizes industry data sets, compares them to literature data sets, and provides guidelines for generating industry-motivated data sets for testing.

Standard literature data sets are defined predominately by three measurements: constraint tightness (τ), the correlation between the coefficients of the objective function and constraints, and the correlation between the constraint coefficients. Additionally, as previously presented in the Chapter 4 on multiple-attributes, the different ranges of coefficients contributes to the difficulty of a problem instance.

A unique trait of industry data is the sparsity of the constraint coefficient matrix. For example, in the manufacturing sourcing problem, this is the case because jobs are not usually routed to every machine. Therefore, the machines where a job does not visit has a zero in the coefficient matrix. This sparsity issue influences most of the data characteristics discussed next.

The four measures of difficulty, constraint tightness, coefficient to objective function correlation, coefficient correlation, multiple attributes, and sparsity are used to characterize the industry data sets.

6.1.1 Constraint Tightness

The constraint tightness ratio, τ, is defined by the equation, $b_j = \tau \sum_{i=1}^{n} A_{ij}$, $\forall j = 1, 2, \ldots, m$, assuming $A_{ij} \geq 0$. In literature test problems, τ is used to define the right-hand-side values, b_j, or the knapsack capacities. Additionally, τ is assumed to be between 0 and 1, and identical for all constraints. It is well documented that as the constraint tightness increases (τ decreases), the problems become more difficult to solve [19, 33].

In literature sets, each constraint is defined by the identical value of τ. However, as evident in the sheet metal data, the more realistic scenario is that the value of τ is different for each constraint. Because each constraint represents capacity on a machine, machines vary in available capacity, and the percentage of a machine's capacity that the sum of the jobs consumes will be different for each machine.

Often a machine has excess capacity and to calculate a value of τ using the actual data would yield a value greater than 1. In this case, all the parts considered for insourcing can be included without restriction from these machines. Until more potential parts are considered, these constraints can effectively be removed. Because of this effect, the number of constraints for the sheet metal data could be reduced from 38 to 10.

To generate industry-motivated data sets, it is assumed that $0 < \tau < 1$. This assumption avoids the need to remove the extra redundant constraints. This also implies that these sets can possess fewer constraints than actual machines in a facility and still be representative of that facility. Additionally, this assumption aligns the industry-motivated data sets with the literature data sets, and allows experimental results of both types to more easily be compared.

6.1.2 Coefficient Correlation

The correlation between the constraint coefficients and the objective function coefficients is considered to be a measure of problem difficulty for both the two- and multi-dimensional knapsack problems [29, 11]. Using the Freville and Plateau [12] method, a correlation factor, K, is defined by the equation $p_j = \left(\sum_{i=1}^{m} A_{ij}\right)/m + Kr_j$, where $A_{ij} \geq 0$ are the constraint coefficients, p_j are the objective function coefficients, and r_j is a random number from $U(0,1)$. However, no method to select or vary K has become standard for test sets. Rather, it is assumed that some correlation should be built into the literature sets to maintain a minimal level of difficulty.

In industry data, because the constraint coefficient matrix is sparse, strong correlation between the constraint coefficients and the objective function coefficients is not likely to exist. Consider the coefficients associated with any one constraint. In the sheet metal data, the average machine is visited by 10% of the parts. Therefore, on average, any one row of the constraint matrix is populated predominately with zeros. These rows with zeros have insignificant correlation to the objective function vector that is populated with positive values. The exception occurs when one of the machines is visited by almost every part. In this case, it is possible to see correlation between the constraint coefficients representing that machine and the objective function coefficients. Hill and Reilly [19] explain that the correlation has an impact on the difficulty when it is extremely negative (close to -1) or, on loose constraints, when the correlation is extremely positive (close to 1). In the sheet metal data, the most highly correlated constraint is -0.05.

Because the correlation in the sheet metal data is nearly zero in every case, and minimal in the rare correlated cases, in generating industry-motivated data sets, the difficulty added by the correlation between the constraint coefficients and the objective functions is not considered as an important factor. As with the constraint tightness factor, this remains consistent with the literature data sets, where no measure of objective function to constraint correlation has emerged as the accepted standard.

6.1.3 Constraint Correlation

The third measure is the correlation between each pair of constraint coefficients. In standard literature data sets, correlation between constraints is not addressed. Hill and Reilly [19] note that a correlation of zero between all pairs of constraint coefficients is common in generating test problems, and that it represents the median case. Hill and Reilly also illustrate that large negative correlation between constraints implies a more difficult problem. This is intuitive in that even if a part contributes significantly to the objective function value and easily fits in one knapsack constraint, if it also consumes the larger part of another knapsack, it is not obvious if the part should be selected or not.

Similar to the correlation between constraints and objective function, in analyzing the sheet metal data, large correlation between constraints is normally non-existent because of the matrix sparsity. It may be the case that many parts have the same routing, and then it would be possible to see a large correlation. However, the sheet metal data indicate that the consumption of a part is proportionally similar for every machine on its routing. That is, a part that is difficult to process on one machine, is difficult on every machine it visits; likewise, a part that is easy to process on one machine, is easy on every machine. Therefore, if correlation exists between constraint coefficients, it is generally positive.

In generating industry-motivated problems, correlation between constraints is not considered a factor. This maintains consistency with literature data sets and with the sparse matrix. However, to account for the possible positive correlation, the proportion of parts that have the same routing is varied. This is defined as the maximum block size and is discussed in more detail in Sections 6.2 and 6.3.

6.2 Industry-Motivated Experimental Data Sets

This section goes into more detail on basic characteristics of industry data and how to generate data sets that reflect those in industry, yet are analogous to the standards generally accepted for literature results. The focus is on the specifics of generating industry-motivated data sets for experimentation. This is followed by the evaluation of the solutions in Section 6.3.

The first characteristic needed to define a data set is the constraint tightness. As discussed above, a constraint tightness factor, τ, is defined between 0 and 1, and is used to calculate the right-hand side values, b. In the sheet metal data, a different value is generated for each constraint. To generate the industry-motivated data sets, the range and gap between τ values is set to be similar to the range of actual values of τ in the sheet metal data. Since each τ value is different it is not straight-forward how to define tighter constraints (a reduced τ). The approach taken is to reduce the τ for each constraint by the same percentage. As will be seen later in the results, the industry-motivated data sets act similar to the literature data sets in that as constraint tightness increases, the problem becomes more difficult.

The other characteristics defining problem difficulty are the correlation between objective function and constraint coefficients, and between constraint coefficients. As addressed in Sections 6.1.3, these correlations are not prevalent in either literature data sets or industry scenarios; thus, these correlations are not applied as factors for defining the industry-motivated sets.

Although the correlation between constraint coefficients and objective function coefficients is not a determining factor for generating industry-motivated data sets, the constraint coefficients still drive the calculation of the objective function coefficients. Constraint coefficients are generated such that the resulting objective function coefficients possess a coefficient of variation (CV) that falls in one of three ranges. This is consistent with the method for the MA-MKP experimental sets in previous chapters. Additionally, it provides a method for comparing the constraint coefficients.

Another measure evident in the sheet metal data sets is the presence of multiple attributes, or the MA-MKP structure. This is applied to industry-motivated data sets by defining different coefficient ranges for each constraint. It is an important characteristic to include because parts consume a different amount of capacity on every machine. For example, a part may take seconds on one machine, but hours on another. Additionally, it is theoretically relevant because the range of coefficients impacts the problem difficulty. In generating the industry-motivated data sets for experimentation, the constraint coefficients are generated using the ranges gathered from the sheet metal data.

A characteristic that is not considered in literature data sets is the interaction between insourcing and outsourcing parts. In the sourcing model, two types of variables exist: insourcing variables that represent parts currently outsourced being considered for insourcing, and the outsourcing variables that are currently in house and being considered for outsourcing. The mathematical problem definition is in Chapter 4, but is repeated here for reference. For the insourcing parts, $x_i = 1$ when part i is insourced, and for the outsourcing parts, $y'_h = 1$ when currently loaded part h remains in-house.

$$\begin{array}{lll}
\text{Maximize} & \sum_{i=1}^{n} p_i x_i - \sum_{h=1}^{k} c_h \left(1 - y'_h\right) & \\
\text{subject to} & \sum_{i=1}^{n} t_{ij}^{m} x_i + \sum_{h=1}^{k} t_{hj}^{m} y'_h \leq b_j^{m} + \sum_{h=1}^{k} t_{hj}^{m} & \forall j = 1, 2, \ldots, m \\
& \sum_{i=1}^{n} t_{ij}^{\ell} x_i + \sum_{h=1}^{k} t_{hj}^{\ell} y'_h \leq b_j^{\ell} + \sum_{h=1}^{k} t_{hj}^{\ell} & \forall j = 1, 2, \ldots, m \\
& x_i \in \{0, 1\} & \forall i = 1, 2, \ldots, n \\
& y'_h \in \{0, 1\} & \forall h = 1, 2, \ldots, k
\end{array}$$

To generate problems that possess interaction between the insourcing and outsourcing parts, the constraint and objective function coefficients of these two types of parts must be considered. First, to prevent the solution where all parts are selected for insourcing, the sum of the insourcing coefficients for each constraint is set such that it exceeds the sum of the outsourcing coefficients. Mathematically, the constraint $\sum_{i=1}^{n} t_{ij}^{m(\ell)} \geq \sum_{h=1}^{k} t_{hj}^{m(\ell)}$ is enforced for both m and ℓ. Second, coefficients within the same constraint are set to a value of similar size. This prevents the effect of one (or a few) jobs consuming all the machine capacity and simplifying the problem too much. Finally, the range of coefficients for insourcing variables is defined such that it overlaps the range of outsourcing variable coefficients. If they do not overlap, an extreme case will be generated where all in-house parts are outsourced and all other parts are insourced, or the opposite, where no parts will be insourced, because parts will never be outsourced to make room for new parts.

The final characteristic is the sparsity of the constraint coefficient matrix. Because a part does not usually visit every machine, the matrix will have many zeros. To quantify this, a maximum routing size is defined. Consider the routing that has the largest proportion of the parts. This proportion of parts is defined as the maximum routing size. For example if 70% of the parts have the same routing, then the maximum routing size is 70. Of the 30% that remain, they are evenly distributed across the other possible routings. The number of machines that a part visits is proportional to the sheet metal data. In the sheet metal data, parts visit approximately 3 of the 10 machines. Therefore, in the 5 machine test problems, the parts each visit 2 machines.

Using these discussed techniques, problem sets that reflect industry data, and still possess interesting characteristics comparable to literature sets, can be generated. The next section looks at the solutions of problems generated using these techniques.

6.3 Solution Characteristics

The initial set of industry-motivated test data, generated using the above described techniques, covers a complete block of problems with three levels of constraint tightness, three coefficient ranges, two routing sizes, three total number of parts, and three ratios of insourc-

Table 6.1: Testing Factors

Tightness	**Range of Coeffs**	**Max Route Size**	**Num Parts**	**Ratio: #in : #out.**
1, 2, 3	1, 2, 3	30, 80	2K, 5K, 8K	1:4, 1:1, 4:1

ing to outsourcing parts. The combination of these scenarios are tested and evaluated with respect to CPU solution time.

Table 6.1 displays the ranges of coefficients for this initial data set. Ten problems are run for each of the 162 scenarios, for a total of 1620 problems. For the test problems, constraint tightness is measured at three levels: tight (1), medium (2), and loose (3). This is consistent with literature sets that are measured with a tightness factor of 0.25, 0.50, or 0.75, respectively. Coefficient ranges are varied across three levels: small (1), medium (2), and large (3). This parallels the three levels of coefficient ranges tested in previous chapters with literature data sets. The coefficient range refers to the range of one constraint, not between different constraints. It is a basic assumption of all the test problems that each constraint has a different range of coefficients, what is earlier defined as MA-MKP. The maximum routing size is defined by the routing that is assigned the most parts. The percentage of parts that have that routing is what is called the maximum routing size. The remaining parts (those not on the routing associated with the maximum routing size) are evenly distributed across the other potential routings. Therefore, in the industry-motivated data sets, two sizes of routings are set: a small maximum routing size (30%), and a large maximum routing size (80%). In the 80% routing size, one routing dominates. In the 30% routing size, all the routings have approximately the same number of parts associated with them. The total number of parts is varied across three values: 2000, 5000, and 8000. Finally, the ratio of insourcing parts to outsourcing parts is tested at three levels: 1:4, 1:1, and 4:1.

The results from running the 162 scenarios defined in Table 6.1 are in Table A-2 in the Appendix. The result under consideration is the time to solve the problem optimally using

CPLEX. The most interesting result to notice is that these industry-motivated problems take considerably less CPU time than the multiple-attribute problems generated in earlier chapters. For example, the 1620 industry-motivated problems, with an average of 5000 parts each, require on average 9.2 CPU seconds for CPLEX to solve the problems optimally. In comparison, problems with 298 parts generated using the standard literature procedure suggested by Freville and Plateau [12] require an average of 1989 CPU seconds. Greater detail on the parameters and solution times for these problems are displayed in Table A-3 in the Appendix. Table 6.2 displays the CPU times for industry-motivated problems and MA-MKP problems. The MA-MKP problems listed here differ from the results in Chapter 4 because they include the outsourcing component. Sets are compared that have similar tightness and range of coefficient states. It is obvious that the industry-motivated problems are easier for CPLEX to solve. Note that the size of the problems are not comparable, as the industry-motivated problems range from 2000 to 8000 parts and the MA-MKP are 298 part problems. This extremely large difference in CPU solution time is likely to be attributed to the sparsity of the coefficient matrix. Therefore, it is expected that the maximum routing size will be an indicator of CPU solution time, and the results of the statistical analysis (discussed below) confirm that this expectation holds.

A second factor to note is that in the solutions, the number of parts insourced is nearly always larger than the number of parts outsourced. This is true regardless of the objective function and constraint coefficients generated. For example, for the 1620 problems, on average 41% of the parts are insourced, while only 5% of the parts currently in the plant load are outsourced, or from the other perspective, 95% remain in-house. As a result, a valid question is, why does the solution procedure select the in-house parts over the outsourced parts if their objective function and constraint coefficients are similar? The driving issue is the capacity for the parts currently in-house. Sufficient capacity exists such that the problem will always be feasible with those parts included in house. Therefore, nothing needs to change and all the constraints are satisfied with the current parts. However, to insource a part currently outsourced requires enough space on each machine on the part's routing, and possibly the

Table 6.2: Solution Time Comparison: Industry-Motivated and MA-MKP Data Sets

Tight-ness	**Range of Coeffs**	**Ind-Motiv CPU secs**	**MA-MKP CPU secs**
1	1	5.5	537.1
	2	22.2	1023.2
	3	42.2	1204.8
2	1	1.7	1645.1
	2	5.4	2993.3
	3	2.1	4307.8
3	1	0.9	953.7
	2	0.7	2172.7
	3	2.2	3063.9
		9.2	**1989.1**

outsourcing of a current in-house part. That is, a part can only be insourced if all of the machines it requires have excess capacity, or currently loaded parts are removed to free up the required capacities. It is easy to see how this eliminates many currently outsourced parts from consideration for insourcing.

As an example of this, consider the following mathematical problem: let x represent the binary decision of whether or not to insource a currently outsourced part, and let y represent the binary decision of whether or not to outsource the part currently produced in house. In this example, both parts visit all three of the machines, and hence are in each constraint.

$$\begin{aligned}
\text{Maximize} \quad & 8x + 7y \\
\text{subject to} \quad & 10x + 10y \leq (0.25)(10 + 10) + 10 = 15 && (6.1) \\
& 3x + 1y \leq (0.25)(3 + 1) + 1 = 2 && (6.2) \\
& 11x + 13y \leq (0.25)(11 + 13) + 13 = 18 && (6.3) \\
& x, y \in 0, 1
\end{aligned}$$

In the example, the available capacities are calculated with a τ value of 0.25. The extra value that is added to the available capacity on the right-hand-side in (6.1), (6.2), and (6.3) is the sum of the outsourcing part coefficients, here the coefficients of y. Therefore, the right-hand-side value represents the available capacity plus the capacity consumed by the current load. Because part y is already being processed in-house, time must be available on each machine for part y. In (6.1) and (6.3) for the currently outsourced part, part x, to be brought in-house, the currently in-house part, part y, cannot be processed; therefore, it effectively must be outsourced. However, in (6.2), even excluding part y does not create enough capacity to handle part x. Even though part x contributes more to the objective function, it is not feasible, because of just one machine constraint, and it cannot be included. This infeasibility will never occur with an insourcing part (e.g., part y) because it is already included in the plant load, which is expressed via the right-hand-side calculation. Therefore, the insourcing parts will be selected more frequently than the outsourcing parts.

To more carefully evaluate the impact of the various factors on solution time, a statistical analysis is applied to the 1620 problems. A complete block of test problems was formed to test 14 different factors. To evaluate the significance of each factor, single- and two-factor ANOVA tests were performed, and the results that possess a statistically significant impact are displayed in the form of p-values in Table 6.3. The p-value is the probability that the correlation seen in the data would have been seen by chance (if no relationship exists between the variables). Therefore, a small p-value (usually below 0.05) implies a statistically significant correlation. To get the p-value, an F-test was performed, and only the significant correlations are presented in the table with their respective p-value. In each section of the table, up to three p-values are shown: the factor in the left column headings, the factor in the top row headings, and the interaction (Inter.) between the two factors.

Evaluating the results, constraint tightness (p=0.0089), maximum routing size (p=0.0086), and the number of parts (p=0.0015) are statistically significant indicators of problem difficulty, as defined as the CPU time required to solve the problem optimally. The influence of constraint tightness is not surprising and is consistent with testing on literature test problems. Figure 6.1(a) displays the solution time for the three levels of constraint tightness.

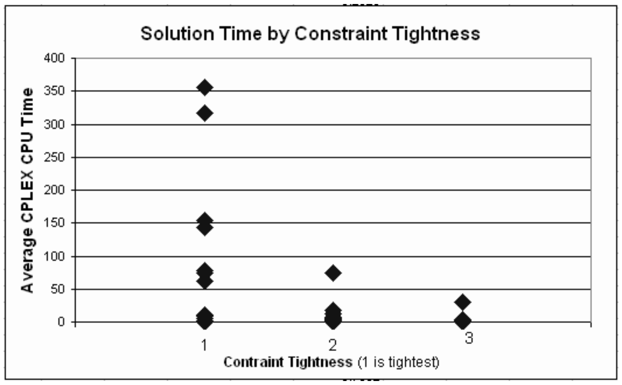
Table 6.3: Industry-Motivated Data ANOVA Test Results

	Constraint Tightness	**Range of Coeffs**	**Max Route Size**	**Num Parts**	**Ratio Parts**
Constraint Tightness	p=0.0089	Tight p=0.0085 *Range not sig.*	Tight p=0.0057 Route p=0.0057 Inter. p=0.0057	Tight p=0.0036 #Parts p=0.0005 Inter. p=0.0004	Tight p=0.0104 *Ratio not sig.*
Range of Coeffs	x	*not sig.*	Route p=0.0083 *Range not sig.*	#Parts p=0.0014 *Range not sig.*	*not sig.*
Max Route Size	x	x	p=0.0086	Route p=0.0046 #Parts p=0.0007 Inter. p=0.0007	Route p=0.0094 *Ratio not sig.*
Num Parts	x	x	x	p=0.0015	#Parts p=0.0018 *Ratio not sig.*
Ratio Parts	x	x	x	x	*Ratio not sig.*

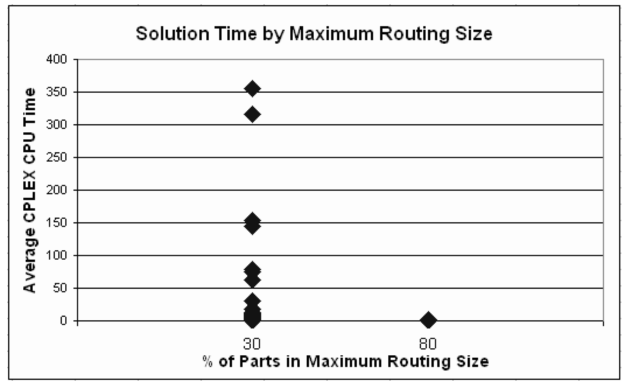
Confirming the p-value, this graph shows an obvious correlation between constraint tightness and solution time.

Maximum routing size is noteworthy because it is an indicator of the coefficient matrix sparsity. As can be seen in Figure 6.1(b), the maximum routing size of 30% appears to be more difficult to solve than problems with a maximum routing size of 80%. Recall that a maximum routing of 80% allows one routing to dominate the problem. However, with the 30% maximum routing, all the routings are similar in size. The graph indicates that the more evenly distributed routing size (30%) is more difficult than the problems with one routing dominating (80%). In literature problems, the constraint matrix contains very few zeros and effectively the routes are all the same size (the size equals the number of parts). Therefore, it is consistent that the industry-motivated problems with routes evenly sized (30%) are more difficult to solve.

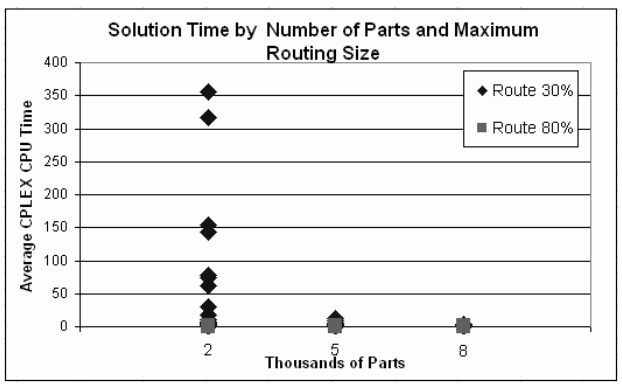
According to the single-factor ANOVA, the lower the number of parts (variables), the greater the time to solve for the optimal solution. This is not intuitive, so it is investigated further by graphing the results. The results from the two-factor ANOVA tests with constraint tightness and maximum routing size are shown in Figures 6.1(c) and 6.1(d). From these graphs, it is clear that the number of parts is not a valid indicator of solution time. Rather, what appears as a correlation between number of parts and solution time is accounted for by the maximum routing size and the constraint tightness factors.



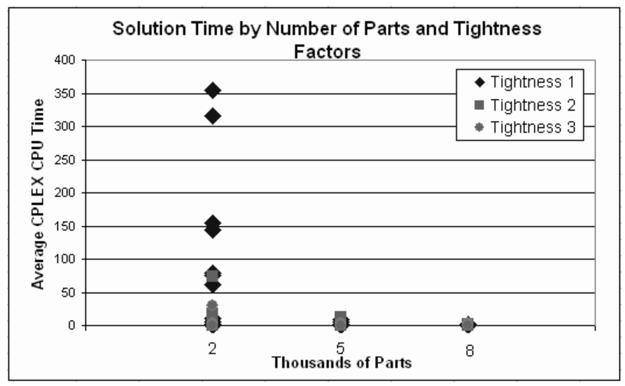
(a) Constraint Tightness



(b) Maximum Routing Size



(c) Number of Parts and Max Routing Size



(d) Number of Parts and Constraint Tightness

Figure 6.1: Statistically Significant Interactions

6.4 Conclusion

The focus of this chapter is a shift from the structure of the problem, to the structure of the data. Data sets are generated and tested that reflect industry data, while maintaining the defining characteristics of literature data sets. The impact of the various features in the industry-motivated data sets are evaluated with respect to solution time. The factors with significant impact on the solution time are the constraint tightness and the maximum routing size.

The critical idea identified in this chapter is that industry-motivated data sets are much easier to solve than the theoretically generated data sets. This is likely to be attributed to the sparse constraint coefficient matrix, and the results from testing the routing size are consistent with that conjecture. Addtionally, sparsity can contribute to a situation where optimal solutions appreciably dominate other solutions, and are therfore easy to identify. This is often the case in industry problems where one must conisider only economically feasible and implementable solutions.

Chapter 7

Multiple-Plant Sourcing Model

In this chapter we extend the sourcing model to include multiple facilities and perform experimentation on problems with two, three, and four facilities. With the recent trend in company acquisitions, it is often the case that a single company has multiple facilities with some overlap in manufacturing capabilities. By shifting the production of parts to different plants, capacity can be better utilized, and the overall cost to the company can be reduced. The multi-plant sourcing problem (MPSP) must determine which parts to produce at each of the plants (or facilities). It is assumed that prior to this point, each plant has made the strategic level decision of which parts must be kept in-house; then, only the remaining parts are considered in the model as eligible for outsourcing.

7.1 Mathematical Formulation

Much like the sourcing problem for a single-plant, the MPSP model effectively outsources all the parts eligible for outsourcing from all the plants into one list. From this list, a part is either selected to be insourced into a facility or outsourced externally. The decision variables, parameters, and mathematical formulation for the MPSP are as follows:

Decision variables:

- $x_{if} = 1$ if part i is selected to be insourced into facility f, 0 otherwise
- $y'_{if} = 1$ if part i remains in facility f, 0 if part i is outsourced from facility f

Parameters:

- E = set of all external parts, or those parts not currently produced at any facility f
- I_f = set of parts from facility f that can be outsourced
- $I = \bigcup_f I_f$ = all possible parts that can be outsourced from the facilities
- $\bar{I}_f = I - I_f$ = all parts that can be outsourced from all facilities besides facility f
- p_{if} = profit associated with insourcing part i into facility f, $p_{if} > 0$
- c_{if} = cost associated with outsourcing part i from facility f, $c_{if} > 0$
- t^{ℓ}_{ijf} = labor time consumed by part i on machine j in facility f
- t^{m}_{ijf} = machine time consumed by part i on machine j in facility f
- b^{ℓ}_{jf} = labor time capacity on machine j at facility f
- b^{m}_{jf} = machine time capacity on machine j at facility f

A preprocessing step is used to adhere to the standard MKP assumptions that $p_{if} > 0$ and $c_{if} > 0$. As discussed in the outsourcing single-plant model, if it is profitable for a facility f to outsource a part i, that is, if $c_{if} \leq 0$, then it is assumed that the part will be outsourced and it is always true that $1 - y'_{if} = 1$. Therefore, there is no outsourcing decision that needs to be made, and the parts can be removed from the outsourcing portion of the problem, leaving only parts with $c_{if} > 0$. Additionally, these removed parts, when $c_{if} \leq 0$, can be added to the set of external parts, E, so that they are still eligible to be insourced into other facilities. Likewise, if $p_{if} \leq 0$, then there is no benefit to consuming the capacity required for that part, and it will never be insourced. Therefore, all parts with $p_{if} \leq 0$ can be removed from the insourcing portion of the problem leaving only the parts with $p_{if} > 0$.

$$
\begin{aligned}
\text{Maximize} \quad & \sum_f \sum_{i \in \bar{I}_f, E} p_{if} x_{if} - \sum_f \sum_{i \in I_f} c_{if} \left(1 - y'_{if}\right) && && (7.1) \\
\text{subject to} \quad & \sum_{i \in \bar{I}_f, E} t^m_{ijf} x_{if} + \sum_{i \in I_f} t^m_{ijf} y'_{if} \leq b^m_{jf} + \sum_{i \in I_f} t^m_{ijf} && \forall j, f && (7.2) \\
& \sum_{i \in \bar{I}_f, E} t^\ell_{ijf} x_{if} + \sum_{i \in I_f} t^\ell_{ijf} y'_{if} \leq b^\ell_{jf} + \sum_{i \in I_f} t^\ell_{ijf} && \forall j, f && (7.3) \\
& \sum_f x_{if} \leq 1 && \forall i \in E && (7.4) \\
& \sum_{f' | f' \neq f} x_{if'} \leq \left(1 - y'_{if}\right) && \forall f, i \in I_f && (7.5) \\
& x_{if}, y'_{if} \in \{0, 1\} && \forall i \in I &&
\end{aligned}
$$

The objective function, (7.1), is made up of two parts, insourcing and outsourcing. The first term, representing insourcing, is the summation over all possible parts that can be insourced by facility f. These parts can be insourced either from another internal facility, $i \in \bar{I}_f$, or from an external company currently producing the part, $i \in E$. A profit, p_{if}, is associated with the selection of part i to be insourced into facility f. The second term in the objective function represents the outsourcing activity. When $1 - y'_{if} = 1$, part i is outsourced from facility f, and a cost, c_{if}, is associated with that decision.

The first two constraints, (7.2) and (7.3), are similar to the constraints in the outsourcing single-plant model. In the single-plant model a constraint of this type existed for all machines, but in the MPSP, a constraint exists for all machines in all the facilities. This constraint effectively outsources all the parts that are considered for outsourcing. From this outsourcing, the freed capacity on machine j in facility f, $\sum_{i \in I_f} t^m_{ijf}$, is added to the original capacity of the machine, b^m_{jf}. Then, the y'_{if} variables represent the decision to insource those same parts back into facility f, or rather to keep those parts produced in the facility where they are currently produced. Similarly, (7.3) refers to labor time at machine j in facility f.

The next constraint, (7.4), is an assignment type constraint for the external parts, where each part can only be insourced into at most one facility. For the internal parts, (7.5) enforces that a part must be outsourced before it can be insourced. When a part is outsourced, or $1 - y'_{if} = 1$, the constraint allows the option to either insource or not to insource the part into any facility besides the facility from which it was outsourced. Simultaneously, in this case the constraint allows only one facility to insource each part because when $1 - y'_{if} = 1$, (7.5) takes on the same form as (7.4). However, if a part remains in its original facility, $1 - y'_{if} = 0$, then the constraint does not allow the part to be insourced, or $x_{if} = 0, \forall f$. The parts in the external set, E, are not considered in this constraint set because they do not need to be outsourced prior to being insourced.

7.2 Experimentation

Experimentation is performed on the multiple-plant model for problems with two, three, and four facilities with the industry-motivated data sets developed in Chapter 6. The two-facility problems are related to the Chapter 6 data sets in the following way: the first single-plant problem instance represents Facility 1, the second single-plant problem instance represents Facility 2, and these two instances combine to form the first two-facility problem. For the second problem, the third and fourth single-plant problems form the two-facility problem, and so on for the remainder of the Chapter 6 industry-motivated data set. Therefore, the size of the problem is doubled with respect to the number of variables considered, but it remains the same size for each individual facility. The data sets for the three- and four-facility problems are generated with the same parameters as the two-facility problems. For each problem, all the facilities are "identical" with respect to the parameters. The data sets for the two-, three-, and four-facility problems each contain 270 problems. The only difference is in the number of variables and constraints.

Table 7.1: Multi-Plant Testing Factors

Tightness (CT)	**Range of Coeffs.**	**Max Route Size**	**Total No. Parts**	**Part Ratio #in : #out**
1, 2, 3	1, 2, 3	30%, 80%	4K, 10K, 16K	1:1, 4:1

As in the single-plant industry-motivated data sets, we analyze the multi-plant problem by solving instances that reflect each factor in Table 7.1. The experimentation includes three levels of constraint tightness (CT), two routing sizes (a measure of sparsity), three total number of parts, and two ratios of insourcing to outsourcing parts. The combinations of these scenarios are tested and evaluated with respect to the CPU solution time. For consistency with the single-plant problem, the total number of parts and the part ratio are presented as separate factors. However, in the multi-plant data sets, the 1:1 part ratio is simply the combination of the 4K and 16K data sets, and the 4:1 data set is the same as the 10K data set.

7.2.1 Two-Facility Results

The results from testing the 54 scenarios represented in Table 7.1 are presented in Table A-4 in the Appendix. (The quantity of scenarios is 54 as opposed to 108 because of the overlap between the Total Number of Parts and the Part Ratio factors.) The result under consideration is the time to solve the problem optimally using CPLEX. It is interesting to note that these problems are much more difficult to solve than the single-plant problems with the same data sets. For example, the single-plant problems took on average 9.2 CPU seconds for the optimal CPLEX solution, where the two-facility problems require an average of 1310.8 CPU seconds. However, this is still considerably faster than the much smaller MA-MKP problems that require on average 1989.1 CPU seconds. Table 7.2 displays the time required to solve the various types of problems. Note that for the values in Table 7.2, the MA-MKP literature problems have 298 variables with 5 constraints, the single-plant industry-motivated

Table 7.2: Solution Time Comparison: Single-Plant Industry-Motivated, Two-Facility Industry-Motivated, and MA-MKP Data Sets

Tight-ness	**Range of Coeffs**	**Single-Plant Ind-Motiv CPU secs**	**Two-Facility Ind-Motiv CPU secs**	**MA-MKP CPU secs**
1	1	5.5	2088.5	537.1
	2	22.2	2365.1	1023.2
	3	42.2	5189.0	1204.8
2	1	1.7	1506.7	1645.1
	2	5.4	21.6	2993.3
	3	2.1	165.1	4307.8
3	1	0.9	5.3	953.7
	2	0.7	452.5	2172.7
	3	2.2	3.6	3063.9
	Average	**9.2**	**1310.8**	**1989.1**

problems range from 2000 variables to 8000 variables, and the two-facility problems double the number of variables with a range of 4000 to 16000.

To assess the significance of each factor, single- and two-way ANOVA tests were performed, and the factors with a statistically significant impact are displayed in the form of p-values in Table 7.3. The p-value is the probability that the correlation seen in the data would have been seen by chance (if no relationship exists between the variables). Therefore, a small p-value (usually below 0.05) implies a statistically significant correlation. To compute the p-value, an F-test was performed, and only the significant correlations are presented in the table with their respective p-value. In each section of the table, up to three p-values are shown: the factor in the left column headings, the factor in the top row headings, and the interaction (Inter.) between the two factors.

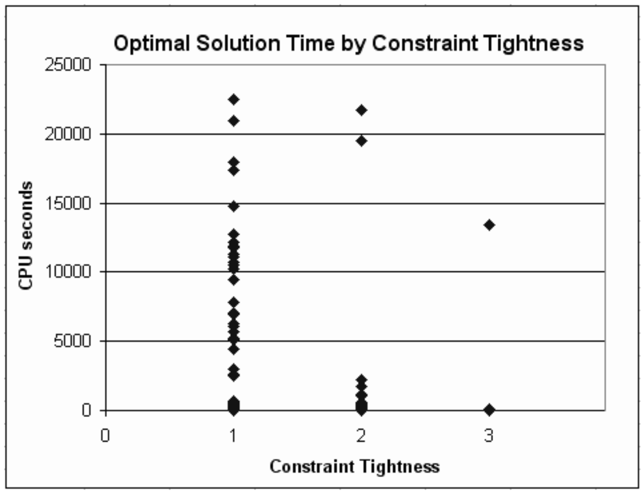
In evaluating the results from the ANOVA test in Table 7.3, three of the five factors stand out as significantly correlated to the solution time: constraint tightness ($p = 0.000000$), total number of parts ($p = 0.000000$), and the part ratio ($p = 0.000000$). We further investigate

Table 7.3: Two-Facility ANOVA Test Results

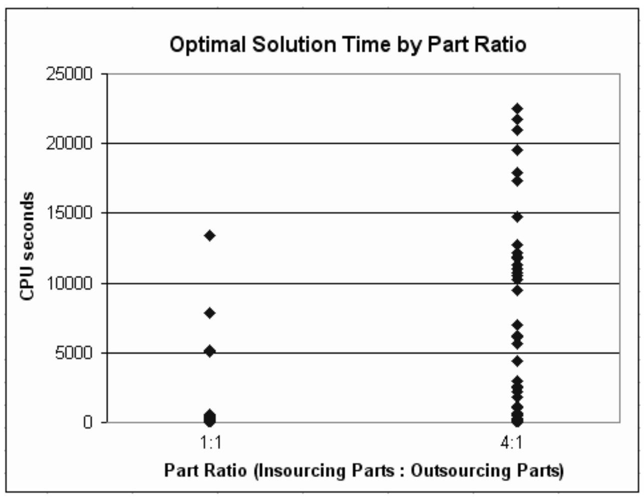
	Constraint Tightness(CT)	**Range of Coeffs**	**Max Route Size**	**Total No. Parts**	**Part Ratio**
Tightness	CT p=0.000000	CT p=0.000000 Inter. p=0.028785	CT p=0.000001	CT p=0.000000 No. Parts p=0.000000 Inter. p=0.000000	Inter. p=0.048094
Range of Coeffs	x			No. Parts p=0.000000	Ratio p=0.000158
Max Route Size	x	x		No. Parts p=0.000000	Ratio p=0.000000
Total No. Parts	x	x	x	No. Parts p=0.000000	
Part Ratio	x	x	x	x	Ratio p=0.000000

these potential correlations by graphing the optimal solution times with respect to the three factors, as shown in Figure 7.1. In each graph, the factor value is expressed along the x-axis value and CPU solution time denotes the y-axis.

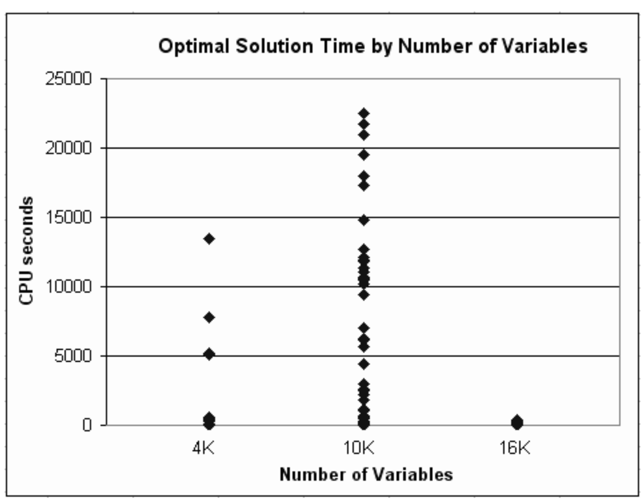
Visually inspecting Figure 7.1, there appears to be a strong correlation between each of the three factors and the optimal solution time. Note that in the constraint tightness and total number of parts graphs, the data sets at each factor level contain the same number of points. For the part ratio graph, the 1:1 ratio has twice as many data points as the 4:1 ratio. Therefore, where it appears that fewer points exist, rather it is because they are overlapping at a solution time of 0. Additionally, the scale for solution time is quite large, as even the first line, 5000 CPU seconds, is quite a long time. Therefore, problems that require much more than 0 CPU seconds are considered difficult as they cannot be solved quickly. Therefore, in the constraint tightness graph, although level 1 has the largest range of values, it is clear that a constraint tightness at level 1 corresponds to the problem difficulty. Most of the long solution times anywhere on the graph are at level 1. Conversely, most of the data points at level 1 have longer solution times than almost all the data points at the other two tightness levels. This result is also evident in Table 7.2 where the long run times are clustered at the constraint tightness level 1. In the part ratio graph, we see a correlation between the 4:1



(a) Constraint Tightness



(b) Part Ratio



(c) Total Number of Parts

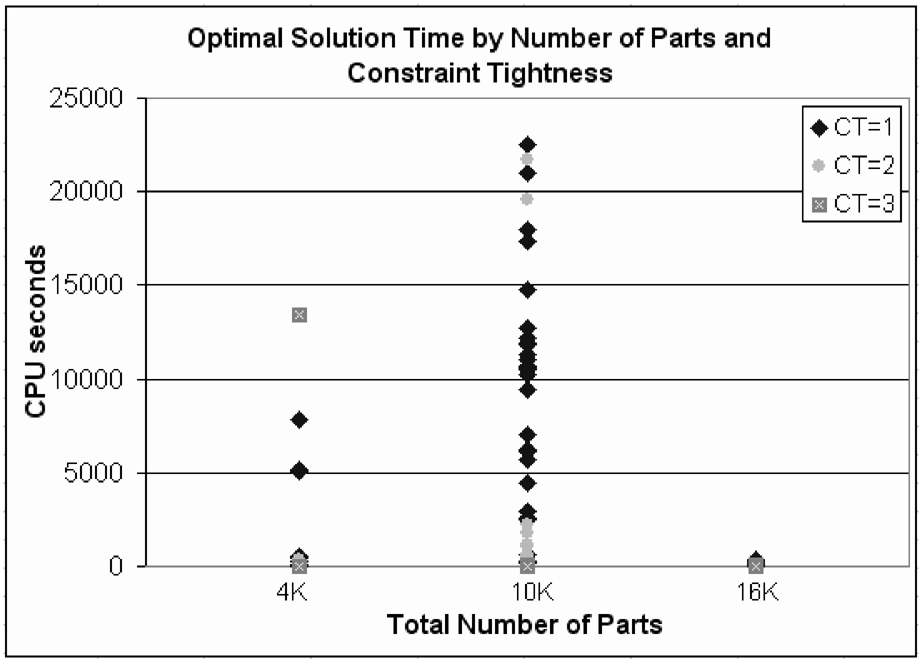
Figure 7.1: Two-Facility Statistically Significant Correlations

ratio and problem difficulty. This result is broken down further in the total number of parts graph. The data points with a 4:1 ratio are the same as the data points with 10K parts. Both the 4:1 ratio and 10K parts appear to be correlated to the solution time. Since the trend on the number of parts graph does not imply a preference for either a small or large number of parts, it is likely that this correlation can be accounted for by either the part ratio factor or another factor.

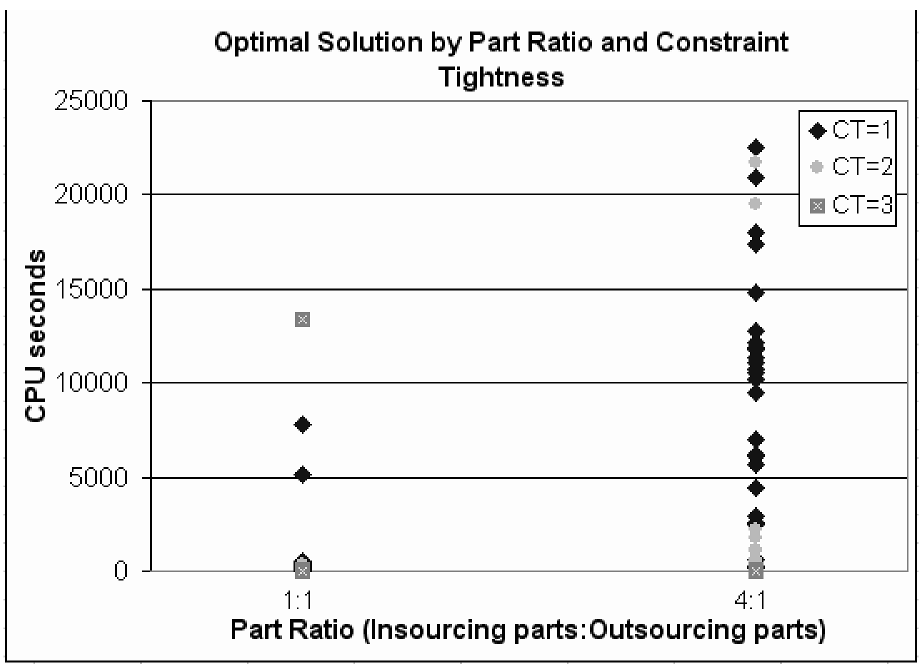
In the single-plant problem, the appearance of a correlation between the number of parts and solution time was a result of the constraint tightness and the maximum route size. Therefore, speculating that the same result will hold true for the two-facility problem, the number of parts and part ratio factors will be evaluated further. In Figure 7.2 the impact of constraint tightness interacting with both the number of parts and the part ratio is presented. From these graphs it is clear that for both the number of parts and the part ratio, the apparent correlation to solution time with the 10K number of parts and the 4:1 part ratio is a result of a constraint tightness factor at level 1. This is not surprising given that in the two-way ANOVA testing, significant interaction occurs between the constraint tightness and both total number of parts and the part ratio.

In Table 7.3 the other two factors with significant interaction are the constraint tightness and the range of coefficients. Additionally, in Table 7.2 at constraint tightness levels 2 and 3, solution times appear to be grouped by the coefficient range. In Figure 7.2.1 the impact of both factors on solution time is presented. The long solution times are distributed across the three coefficient ranges, but with most of the large solution times associated with problems at constraint tightness level 1.

Because the constraint tightness appears to be the significant factor, we take a closer look at results grouped by this factor, and present the solution results and run times in Table 7.4. Constraint tightness is defined at three levels (1, 2, and 3), with level 1 generating the tightest constraints. The column representing the optimal solution time (labeled Opt CPU time), combined with the previous analysis, clearly demonstrates that problems are more difficult



(a) Number of Parts and Constraint Tightness



(b) Part Ratio and Constraint Tightness

Figure 7.2: Two-Facility Impact of Constraint Tightness

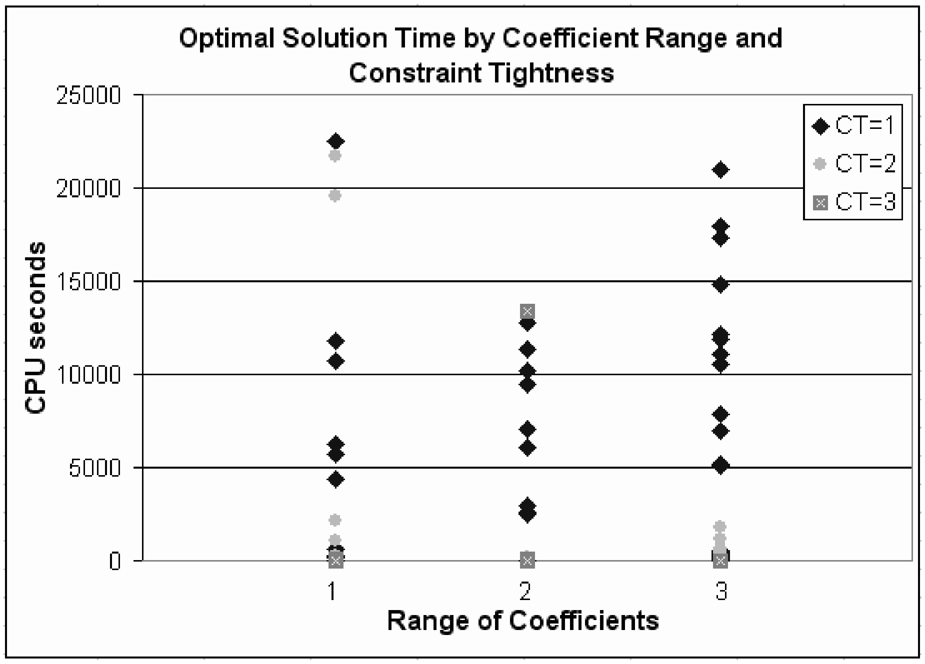


Figure 7.3: Two-Facility Interaction of Constraint Tightness and Range of Coefficients

to solve when constraints are tight. Problems at the tightest constraint level (level 1) require on average 2989 CPU seconds to solve the problem versus a still relatively slow 553 seconds at level 2, and a much faster 7 seconds at level 3. This result that problems with tighter constraints are more difficult is consistent with the ANOVA testing, the examination of the graphs earlier in this section, as well as the MKP literature.

Table 7.4: Two-Facility Solution Results by Constraint Tightness Level

CT Level	Opt CPU time	% In-sourced Fac 1	% Out-sourced Fac 1	% In-sourced Fac 2	% Out-sourced Fac 2
1	2989	37%	38%	37%	38%
2	553	39%	38%	39%	37%
3	7	37%	35%	37%	35%

Besides solution time, note that the percentage of parts moving in and out of each facility is roughly the same for each constraint tightness level. Approximately 35-38% of the current plant load is outsourced to the other facility or to an outside supplier, and 37-39% of the potential parts are insourced into each facility. These results are different from the single-plant model, where we observed a preference for the current load.

Because the two-facility problem is not easy to solve (compared to the single-plant problem) the heuristic, presented in Chapter 5, is run to determine if it is a valuable solution procedure for the industry-motivated problems. The results are displayed in Table 7.5. The heuristic is fast, requiring on average 0.80 CPU seconds, and is accurate, with an average 0.06% gap between the solution and the optimal solution. This gap is smaller than we saw with the MA-MKP problems in Chapter 5 where the average gap was 1.39% and 0.69% for the two data sets. The results from this experiment are consistent with the observed trend that as the problem size increases, the gap decreases. The problems in Chapter 5 have respectively 200 and 298 parts, where the two-facility problems have between 4000 and 16000 parts.

The heuristic performs well despite that it is slightly biased. Because a part can only be allocated to one facility, one of the two facilities has the priority. The heuristic is executed by ranking the extra parts (those not included in the feasible LP solution) and then attempting to add the parts in order of the ranking. For the the two-facility problem, one facility attempts to add all the parts, followed by the second facility attempting to add all the parts. If the part is already added in the first facility, it is no longer available for the second facility.

In summary, the multi-plant model is tested on two-facility problems using the industry-motivated data sets from Chapter 6. The results are consistent with the single-plant problem and MKP literature in that constraint tightness is an indicator of solution time. The tighter the constraints, the more difficult the problem is to solve. Additionally, other apparent correlations between the number of parts and part ratio can be accounted for by the constraint tightness. On average, the two-facility problem is much more difficult to solve (with respect to solution time) than the single-plant problem. However, the heuristic presented in Chapter 5 performs well with a gap of 0.06% between the optimal solution and the heuristic solution. In the single-facility problem, the maximum routing size was a significant factor in

Table 7.5: Two-Facility Heuristic Results

Coeff Range	Tight-ness	Heur. CPU	Optimal CPU	Soln Gap
1	1	0.75	2088.50	0.10%
2	1	0.82	2365.10	0.07%
3	1	0.87	5189.00	0.11%
1	2	0.74	1506.70	0.07%
2	2	0.81	21.60	0.03%
3	2	0.85	165.10	0.05%
1	3	0.72	5.30	0.04%
2	3	0.79	452.50	0.01%
3	3	0.80	3.60	0.04%
	Average	**0.80**	**1310.8**	**0.06%**

determining solution time; however, in the two-facility problem, the maximum routing size does not appear to be correlated to the problem difficulty. Also, unlike the single-facility problem, a similar quantity of parts are sourced in and out of each facility. This could be a result of the data generation technique where the parts in both facilities look very similar because they are generated with the same factors and distributions, or it could mean that the phenomenon we observed in Chapter 6 is not present in multi-plant problems.

7.2.2 Three-Facility and Four-Facility Results

Experimentation similar to the two-facility problem is performed on problems with three and four facilities. Problem instances that reflect the characteristics listed in Table 7.1, with adjustments to the number of parts, are analyzed. Instead of the total number of parts at 4K, 10K, and 16K, the number of parts includes 9K, 12K, and 18K for the three-facility problem, and 12K, 24K, and 32K for the four-facility problem. For the two-, three-, and four-facility problems, the number of parts that can be outsourced from each facility is the same for each scenario, but as the number of facilities increases, the number of parts that can be insourced increases as more parts become available with the additional facilities. For example, in the two-facility problem, if both plants outsource 1000 parts, then both plants can potentially insource the 1000 parts outsourced from the other facility, plus any external parts. In the three-facility problem, if all three plants outsource 1000 parts, then the potential parts to insource into each facility increases to 2000 plus the external parts.

The result under consideration is the time to solve the problem optimally using CPLEX. Table 7.6 summarizes the results from testing the industry-motivated data sets with one, two, three, and four facilities, as well as the MA-MKP data set discussed in Chapter 4. The detailed results from testing the 54 scenarios are listed in Tables A-6 and A-7 in the Appendix.

Table 7.6 displays a trend that with more than one facility, the solution time decreases as the number of facilities increases. A possible explanation for this observation is that with more facilities, the variance of the coefficients for the same part at different facilities is likely to increase, and thus, the decision of how to allocate production will be easier based on a facility dominating for each part in the set.

Table 7.6: Solution Time Summary

Tight-ness	**Coeff. Range**	**Single-Plant**	**Two-Facility**	**Three-Facility**	**Four-Facility**	**MA-MKP**
1	1	5.5	2088.5	7.7	10.2	537.1
	2	22.2	2365.1	8.2	12.3	1023.2
	3	42.2	5189.0	312.5	47.8	1204.8
2	1	1.7	1506.7	2.2	4.2	1645.1
	2	5.4	21.6	1.0	1.5	2993.3
	3	2.1	165.1	5.6	5.2	4307.8
3	1	0.9	5.3	2.7	7.5	953.7
	2	0.7	452.5	47.5	22.0	2172.7
	3	2.2	3.6	2.1	8.8	3063.9
		9.2	**1310.8**	**43.3**	**13.3**	**1989.1**

In analyzing Table 7.6, two rows stand out as more difficult than the others. These are the problems with a tightness level of 1 and a coefficient range of 3, and with a tightness level of 3 and coefficient range of 2. These two problem sets are examined in more detail to see if the cause of these difficult problems can be discovered.

The two rows in Table 7.6 that appear to be most difficult are displayed in Table 7.7 in greater detail with the additional maximum route size category. Looking at these four problem sets for both the three- and four-facility problems, the first row, with a tightness level at 1, coefficient range at 3, and a maximum route size of 30 is more difficult than the other combinations. The results are consistent with literature and earlier findings. The tightness level at 1, indicating the tightest constraints, is a notable indicator of difficulty with literature MKP problems. Additionally, tightness is an indicator of difficulty in our experimentation with the multiple-attribute MKPs in Chapter 4, as well as the single- and two-facility problems. Second, the coefficient range of 3, the largest range of coefficients, is an indicator of difficulty for standard single-dimensional knapsack problems. Finally, with the single-facility problems in Chapter 6 we found a correlation between a maximum route size of 30 and the difficult problems.

Table 7.7: Breakdown of Difficult Three- and Four-Facility Problems

Tight-ness	**Coefficient Range**	**Max Route**	**Avg Opt CPU Time**
Three-Facilities			
1	3	30	590.2
		80	34.9
3	2	30	9.5
		80	85.6
Four-Facilities			
1	3	30	63.5
		80	32.0
3	2	30	18.1
		80	25.8

To complete the multi-plant evaluation, a statistical analysis is performed for both the three- and four-facility problems. As with the single-plant and two-facility problems, an one- and two-way ANOVA tests are performed on the full block of parameters. The factors with a statistically significant impact are displayed in the form of p-values in Tables 7.8 and 7.9. The p-value is the probability that the correlation seen in the data would have been seen by chance (if no relationship exists between the variables). In each section of the table, up to three p-values are shown: the factor in the left column headings, the factor in the top row headings, and the interaction (Inter.) between the two factors.

Table 7.8: Three-Facility ANOVA Test Results

	Constraint Tightness(CT)	**Range of Coeffs**	**Max Route Size**	**Total No. Parts**
Tightness (CT)	CT p=0.002573	CT p=0.001124 Range p=0.001898 Inter. p=0.000006	CT p=0.000001 Route p=0.044817 Inter. p=0.002449	CT p=0.002041 No. Parts p=0.092791 Inter. p=0.021274
Range of Coeffs	x	Range p=0.004147	Range p=0.003135 Route p=0.044981 Inter. p=0.001850	Range p=0.003318 Inter. p=0.016121
Max Route Size	x	x		
Total No. Parts	x	x	x	

Table 7.9: Four-Facility ANOVA Test Results

	Constraint Tightness(CT)	**Range of Coeffs**	**Max Route Size**	**Total No. Parts**
Tightness (CT)	CT p=0.000144	CT p=0.000057 Range p=0.010086 Inter. p=0.000055	CT p=0.000140	CT p=0.000124
Range of Coeffs	x	Range p=0.018218	Range p=0.018504	Range p=0.017668
Max Route Size	x	x		
Total No. Parts	x	x	x	

Although both ANOVA tests display statistical significance between tightness and solution difficulty and between the coefficient range and solution difficulty, when the individual and interaction between two factors are graphed, no single level of any one factor appears to be an indicator of solution difficulty.

Finally, in the single-plant experimentation, a preference for in-house parts was observed. This did not occur for the two-facility problem and on average is not evident in the three- or four-facility problems. However, looking at the movement of parts by the range of coefficients reveals that in the four-facility problem, as the range of coefficients decreases, there exists an increasing preference for in-house parts. Table 7.10 displays the average movement in and out of the facilities by coefficient range for the four-facility problem. At the smallest coefficient range, range 1, a smaller percentage of possible parts are outsourced. This result is intuitive because the tighter the range of coefficients for a constraint, the less likely that it will be worth it to remove a part from the current load. That is, “boulder” parts, that free up large amounts of capacity to allow more profitable parts to be insourced, are not available at the tightest coefficient ranges because all the coefficients are within a small range.

Table 7.10: Four-Facility Part Movement

Coeff. Range	**% In**	**% Out**
1	19%	8%
2	30%	49%
3	38%	78%

In summary, experimentation is performed on three- and four-facility problems with industry-motivated data sets. In evaluating the multi-plant problem in general, as the number of facilities increases, the problem becomes easier to solve. This indicates that with the increase in facilities, and hence increased variation of coefficients, it is more likely that one solution will dominate. Unlike the single-plant and two-facility problems, with the three- and four-facility problems, no single factor stands alone as an indicator of problem difficulty. Additionally, the preference for in-house parts found in the single-plant problem is only visible for the four-facility problems with the smallest range of coefficients.

Chapter 8

Industry Experience

As part of a two-year project, we worked with personnel of a large U.S.-based manufacturer of refrigerated display cases. Such a product is designed to use many sheet metal parts. The company has five sheet metal facilities and is part of a parent organization that currently manufacturers other products that utilize purchased sheet metal parts from outside the organization. The primary objectives of the project were to more effectively utilize the sheet metal processing capacity at all the sheet metal facilities and to provide sheet metal parts at a lower cost within the company. To aid in this decision-making, we created decision-support tools to evaluate insourcing opportunities at five sheet metal facilities. One of the manufacturing facilities was a large, state-of-the-art sheet metal fabrication facility (possibly the largest installation in the U.S.) and we focused on that plant first (henceforth referred to as "SMFF," for sheet metal fabrication facility).

After an initial analysis of possible opportunities, it was determined that insourcing parts from two locations within the organization to SMFF should be comprehensively evaluated (no outsourcing was considered). To that end, 1857 part types were analyzed and their data provided for the insourcing decision-support tool. Data for this analysis included the current price paid for the part types. Other data included manufactured parts specifics, materials costs, processing times, transportation costs ($0.03/lb), etc. Also included were facility overhead costs and a 10% markup rate on the estimated manufactured cost (to cover small errors in the data). The decision-support tool calculated the price SMFF would charge the other companies within the organization for each part and would not insource any part that could not be provided at a price lower than the current price.

The data collection phase of the project was particularly challenging due to the thousands of parts that were ultimately classified into various part types based on the machines that were visited. Also, throughout the plant, many discussions related to fixed costs, overhead, opportunity costs and the like were spirited and forced many to reexamine SMFF's cost structure. In the end, all results had to be presented in the two ways we use below to accommodate opposing views on SMFF's cost structure.

Of the parts that could be provided at a price lower than the current price, the decision-support tool implicitly evaluates all possible combinations of adding parts with the objective of adding those parts that maximize profit. Profit is defined as the total revenue minus the maximum costs incurred (material costs, transportation costs, overhead costs, full labor costs, and markup). In reality, there is additional profit since there is labor and overhead already being charged to SMFF that may be better utilized if additional parts are insourced. The decision to add a part is constrained by the capacity of each machine type in the system. Scenarios were run that included 15, 20, and 25 working days per month as well as low, medium and high facility loads.

The final model had 70,566 variables (1857 part types multiplied by 38 constraints), 19 machines, and was implemented in Excel using the Premium Solver from Frontline Systems. Visual Basic for Applications was used to program the graphical user interface. Since our results from Section 6.3 indicate that industry-motivated data sets are significantly easier to solve than problems attempted in the literature (which are on the order of 400 variables in size), we hypothesized that a heuristic would not be necessary to solve these much larger problems. In actuality, to reduce runtime, we solved these large problems by separating the part types into two batches and then running a combined problem with the part types selected from each of the two batches.

In the final analysis, 1644 of the 1857 part types were recommended to be added to the load at SMFF. The conservative final financial results are summarized below:

- Profit to SMFF (10% markup rate; no adjustment for labor and overhead): \$16,299/month
- Savings to Customers: \$261,977/month
  - Total Savings to parent organization: \$278,276/month (or \$3.34M/year)

The final financial results including the impact of considering the underutilized machine capacity and labor are summarized below:

- Profit to SMFF (10% markup rate; no adjustment for labor and overhead): \$16,299/month
- Labor Costs previously charged and can now be utilized: \$34,809/month
- Overhead Costs previously charged and can now be utilized: \$33,397/month
- Savings to Customers: \$261,977/month
  - Total Savings to parent organization: \$346,482/month (or \$4.16M/year)

The benefits of the project were mani-fold. The primary benefit was in providing a systematic and quantitative process for evaluating decisions related to sheet metal fabrication capacity. The project also provided an analysis of the current machine utilizations, which initiated quite a few "accelerated change programs" (the company's name for Kaizan Events) in an attempt to break bottlenecks. Multiple scenarios to be analyzed by the decision-support tool were constructed to aid in focusing resources. The project lead naturally to a company-wide reevaluation of sourcing decisions, which was conducted as part of the strategic planning process. The final result is that SMFF will continue to be a focal point of sheet metal fabrication within the organization and the facility is in a stronger competitive position.

Our work with the SMFF and the parent organization motivated us to create an Excel-based tool to solve the single-plant general sourcing problem. The development of this tool was funded by the Center for High Performance Manufacturing (CHPM) at Virginia Tech, a member-based research center to which the parent organization belongs. To ensure widespread usability of the tool, it incorporates a free open source LP and MIP solver, LPsolve, created by Michel Berkelaar and maintained by Sam Buttrey. LPsolve can be downloaded from the yahoo group page at `http://groups.yahoo.com/group/lp_solve/files/`. The Excel-based tool is proprietary software of the CHPM and its member companies. The CHPM should be contacted for further information on this tool at `http://chpm.ise.vt.edu/`.

Chapter 9

Conclusions and Future Research

This dissertation covers research contributions in three main areas. First, the tactical level sourcing decision was defined and modeled as a multidimensional knapsack problem (MKP). Second, characteristics of the sourcing problem that differ from standard MKP literature problems were identified and tested with respect to their effect on problem difficulty. This includes the multiple-attribute classification and generation techniques for industry-motivated data sets. Finally, the model was extended to include multiple facilities allowing for movement of parts between facilities.

The first research contribution covers modeling sourcing decisions as a multidimensional knapsack problem (MKP). The sourcing decision was defined as, given the current machine and labor capacity, selecting from a list of currently outsourced parts, which parts to insource, and which parts from the current plant load should be outsourced. Extensions to the basic model considered multiple time periods, the option of increasing the available labor, and the multi-plant model.

Secondly, characteristics of the sourcing problem were identified and compared to standard MKP test problems. The first characteristic noted was the multiple-attribute (MA) structure. Experimental results demonstrated that problems with the MA-MKP structure are significantly more difficult to solve than the standard literature problems with the single-attribute structure. For small problems, with 100 variables and 5 constraints, the MA-MKP required on average 206 CPU seconds to find the optimal solution, versus 6.4 for the SA-MKP problems. Additionally, although experimentation with standard MIP solution techniques was somewhat effective in reducing the time required to solve the problem to optimality, none of the techniques brought the solution time close to the level of the SA-MKP solution times. Therefore, a heuristic was developed to solve the MA-MKP. It is a simple heuristic that begins with the LP solution and uses list processing rules to improve and maintain feasibility. Although simple, the heuristic is effective and the solutions improved as the size of the problem increased. For example, the average gap between the heuristic solution and the optimal solution was 1.39% for the 200-part problem and was reduced to 0.69% when the size of the problem increased to 298 parts.

Other characteristics of the sourcing problem that differ from standard literature MKP problems were identified through evaluation of actual industry data. The major differences between industry problems and MKP literature test problems are a result of the fact that in most industry problems parts do not visit every machine. In the industry data, a part routing contained roughly one-third of the machines. Therefore, two-thirds of the constraint matrix contains zeros, compared to the completely dense constraint matrix considered in the literature. Industry-motivated data sets were generated that reflect the industry data, yet have a form similar to the literature data sets to allow for comparisons.

Testing the industry-motivated data sets illustrated which factors are significant with respect to algorithm runtime (most notably, constraint tightness and maximum route size). The testing also identified an interesting *phenomenon* that did not occur with standard data sets; that is, the preference for parts currently in the plant load, or in other words, the tendency to outsource very few parts from the current plant load. This phenomenon coincides with our experiences in industry, which when originally observed, we speculated was due to the decision-makers being conservative when it came to outsourcing from their plant. We now see that it was due to how companies build a solution that is so highly constrained that it is difficult for parts from the outside to fit into it. This is a topic we plan to pursue in our future research.

Finally, within the third research contribution, the sourcing model was extended to include multiple facilities. In this model, the objective function takes into consideration more than one facility at a time. The decisions in the multiple-facility model are how to allocate parts across the various facilities. Experimentation was conducted on the two-facility model and results pertaining to problem difficulty were consistent with the single-plant model in that constraint tightness is an indicator of problem difficulty. However, compared to the single-plant model, which required on average 9.2 CPU seconds to solve optimally, the two-facility problem required on average 1310.8 CPU seconds. Additionally, the earlier noted phenomenon where a facility has a preference for in-house parts, was not observed in the two-facility model. This could be the result of the data generation technique or that the phenomenon is not present in the multi-plant problem; however, we plan to pursue this topic in future research.

Multi-plant experimentation on three- and four-facility problems revealed that as the number of facilities increases, the problem is easier to solve. Compared to the 1310.8 CPU seconds required to solve the two-facility problem, the three- and four-facility problems require 43.3 and 13.3 CPU seconds, respectively. A likely explanation for the decrease in difficulty is that with additional facilities, one facility is more likely to be the obvious solution for each part and sub-optimal solutions are quickly eliminated. Additionally, with the three- and four-facility problems, no single factor emerged as an indicator of problem difficulty, and on average, as with the two-facility problem, the preference for in-house parts does not exist. Understanding why this phenomenon is present with the single-plant problem and not with the multi-plant problems is a topic to be studied in future research.

Our experiences from a two-year project with a large U.S.-based manufacturer of refrigerated display cases were reported. In addition to providing a foundation from which to study this important tactical sourcing problem, results from the corresponding decision-support tool were used in formulating the company's manufacturing strategy and saved the parent organization up to $4.16M per year.

Future research will involve further extension of the MA-MKP application area beyond the sourcing problem. There are many applications that could prove to be interesting areas of research. For example, consider a company like Valassis. Valassis prints coupon inserts for Sunday newspapers. Valassis has accounts setup with many companies in different market segments (e.g., they have accounts with both Pampers and Huggies in the diaper segment, Pizza Hut and Papa Johns in the take-out pizza segment, etc.). The coupon insert can be considered a knapsack and Valassis will solve this problem over a planning horizon, which means that there are multiple knapsacks to consider simultaneously. Each knapsack will have a size constraint expressed by the square inches of insert area. There will also be constraints associated with how often to run each coupon insert over the planning horizon, including different account preferences (e.g., Pampers likes to be included in the coupon insert for the first Sunday of each month). An additional constraint that will make the MKP interesting for this application area is that Valassis is not permitted to run coupons from the same market segment in the same week.

Other examples of application areas include a company (like Pizza Hut) that has to decide which type of coupon to allow a company like Valassis to include based on what products, services, or times they want customers to consume. A similar application is in the apparel industry, where the decision of which sales to run is based on the need to free up store capacity for new, more profitable products. Additional applications can be found in problems dealing with the of allocation of people, products, or services. For example, in the health care industry, lab technicians and other experts are allocated to hospitals not only to fulfill demand, but in a way that maximizes profit. Similarly, the allocation of projects to teams in a consulting firm could be modeled as an MKP. Such a problem begins to cross into the realm of scheduling. Since industry problems can be solved quickly, it might be possible to integrate this model within a scheduling algorithm for a problem such as scheduling jobs to cells in a cellular manufacturing facility. It is our intent to explore new application areas for sourcing problems modeled as MKPs in our future research.

Bibliography

[1] Balas, E., "An Additive Alogrithm for Solving Linear Programs with Zero-One Variables," *Operations Research*, 13, 4, 517–549 (1965).

[2] Balas, E., and Martin, C., "Pivot and Complement-A Heuristic for 0-1 Programming," *Management Science*, 26, 86–96 (1980).

[3] Bertsimas, D., and Demir, R., "An Approximate Dynamic Programming Approach to Multidimensional Knapsack Problems," *Management Science*, 48, 550–565 (2002).

[4] Cãnez, L.E. Platts, K., and Probert, D., "Developing a Framework for Make-or-Buy Decisions," *International Journal of Operations & Production Management*, 20, 11, 1313–1330 (2000).

[5] Chu, P., and Beasley, J., "A Genetic Algorithm for the Multidimensional Knapsack Problem," *Journal of Heuristics*, 4, 63–86 (1998).

[6] CPLEX Division, ILOG Inc., 889 Alder Avenue, Suite 200, Incline Village, NV, *Using the CPLEX Callable Libraries Version 6.0* (1998).

[7] Dammeyer, F., and Voß, S., "Dynamic Tabu List Management Using Reverse Elimination Method," *Annals of Operations Research*, 41, 31–46 (1993).

[8] Dekkers, R., "Strategic Capacity Management: Meeting Technological Demands and Performance Criteria," *International Journal of Production Research*, 40, 15, 3895–3911 (2002).

[9] Drexl, A., "A Simulated Annealing Approach to the Multi-Constraint Zero-One Knapsack Problem," *Computing*, 40, 1–8 (1988).

[10] Everett III, H., "Generalized Lagrange Multiplier Method for Solving Problems of Optimum Allocation of Resources," *Operations Research*, 2, 399–417 (1963).

[11] Fréville, A., "The Multidimensional 0-1 Knapsack Problem: An Overview," *European Journal of Operational Research*, 155, 1–21 (2004).

[12] Fréville, A., and Plateau, G., "Hard 0-1 Multiknapsack Test Problems for Size Reduction Methods," *Investigacion Operativa*, 1, 251–270 (1990).

[13] Gabrel, V., and Minoux, M., "A Scheme for Exact Separation of Extended Cover Inequalities and Application to Multidimensional Knapsack Problems," *Operations Research Letters*, 30, 252–264 (2002).

[14] Garey, M., and Johnson, D., *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, San Francisco (1979).

[15] Gavish, B., and Pirkul, H., "Efficient Algorithms for Solving Multiconstraint Zero-One Knapsack Problems to Optimality," *Mathematical Programming*, 31, 78–105 (1985).

[16] Gilmore, P., and Gomory, R., "The Theory and Computation of Knapsack Functions," *Operations Research*, 14, 1045–1075 (1966).

[17] Glover, F., and Kochenberger, G., "Critical Event Tabu Search for Multidimensional Knapsack Problems," *Meta-Heuristics: Theory and Applications*, pp. 407–427 (1996).

[18] Hanafi, S., and Fréville, A., "An Efficient Tabu Search Approach for the 0-1 Multidimensional Knapsack Problem," *European Journal of Operational Research*, 106, 659–675 (1997).

[19] Hill, R., and Reilly, C., "The Effects of Coefficient Correlation Structure in Two-Dimensional Knapsack Problems on Solution Procedure Performance," *Management Science*, 46, 2, 302–317 (2000).

[20] Insinga, R., and Werle, M., "Linking Outsourcing to Business Strategy," *Academy of Management*, 14, 4, 58–70 (2000).

[21] Jennings, D., "Outsourcing Opportunities for Financial Services," *Long Range Planning*, 29, 3, 393–404 (1996).

[22] Jeroslow, R., and Smith, T., "Experimental Results on Hillier's Linear Search," *Mathematical Programming*, 9, 371–376 (1975).

[23] Lee, J., and Guignard, M., "An Approximate Algorithm for Multidimensional Zero-One Knapsack Problems - A Parametric Approach," *Management Science*, 34, 402–410 (1988).

[24] Lorie, J., and Savage, L., "Three Problems in Capital Rationing," *Journal of Business*, 28, 229–239 (1955).

[25] Loulou, R., and Michaelides, E., "New Greedy-Like Heuristics for the Multidimensional Zero-One Knapsack Problem," *Operations Research*, 27, 6, 1101–1114 (1979).

[26] Luenberger, D., "Quasi-Convex Programming," *Siam Journal of Applied Mathematics*, 16, 1090–1095 (1968).

[27] Magazine, M., and Oguz, O., "A Heuristic Algorithm for the Multidimensional Zero-One Knapsack Problem," *European Journal of Operational Research*, 16, 319–326 (1984).

[28] Mansini, R., and Speranza, M., "A Multidimensional Knapsack Model for Asset-Backed Securitization," *Journal of the Operational Research Society*, 53, 822–832 (2002).

[29] Martello, S., and Toth, P., *The 0-1 Knapsack Problem in Combinatorial Optimization*, John Wiley and Sons, New York.

[30] Momme, J., "Outsourcing Manufacturing to Suppliers - A Practical Framework," In *The Fourth SMESME International Conference*. Aalborg University, Denmark (May 2001).

[31] Nemhauser, G., and Ullmann, Z., "Discrete Dynamic Programming and Capital Allocation," *Management Science*, 15, 494–505 (1969).

[32] "OR-library," http://mscmga.ms.ic.ac.uk/jeb/orlib/mknap-info.html.

[33] Pirkul, H., "A Heuristic Solution Procedure for the Multiconstraint Zero-One Knapsack Problem," *Naval Research Logistics*, 34, 161–172 (1987).

[34] Senju, S., and Toyoda, Y., "An Approach to Linear Programming with 0-1 Variables," *Management Science*, 15, 196–207 (1968).

[35] Shih, W., "A Branch and Bound Method for the Multiconstraint Zero-One Knapsack Problem," *Journal of the Operational Research Society*, 30, 369–378 (1979).

[36] Soyster, A.L. Lev, B., and Slivka, W., "Zero-One Programming with many variables and few constraints," *European Journal of Operational Research*, 2, 195–201 (1977).

[37] Toyoda, Y., "A Simplified Algorithm for Obtaining Approximate Solutions to Zero-One Programming Problems," *Management Science*, 21, 1417–1427 (1975).

[38] Vankatesan, R., "Strategic Sourcing: To Make or Not to Make," *Harvard Business Review* (November-December 1992).

[39] Vasquez, M., and Hao, J., "A Hybrid Approach for the 0-1 Multidimensional Knapsack Problem," *RAIRO; Operations Research*, 106, 659–675 (2001).

[40] Volgenant, A., and Zoon, J., "An Improved Heuristic for the Multidimensional 0-1 Knapsack Problems," *Journal of the Operational Research Society*, 41, 963–970 (1990).

[41] Weingartner, H., *Mathematical Programming and the Analysis of Capital Budgeting Problems*, Prentice-Hall, Englewood Cliffs, NJ (1963).

[42] Weingartner, H., and Ness, D., "Methods for the Solution of the Multidimensional 0/1 Knapsack Problem," *Operations Research*, 15, 83–103 (1967).

Appendix

The parameters presented in Table A-1 are defined to generate the industry-motivated data sets analyzed in the experimentation in Section 6.3. For each of the 162 data sets, 10 problems are generated, for a total of 1620 problem instances.

The total number of variables and constraints is normally used to define the problem size of an MKP. We define the number of variables with respect to both the number of insourcing variables and the number of outsourcing variables, labeled respectively, "# In Vars" and "# Out Vars," in Table A-1. Also, for each of the problems, 5 machines are considered. This is because in the sheet metal data, of the 38 constraints (19 machines), only 10 constraints (5 machines) were determined to be critical to the problem.

Constraint coefficients are generated from a uniform distribution over a defined range that is different for each constraint. The next 10 columns, under the label "Maximum of Coefficient Range," represent the maximum value of the range for each constraint and each type of variable (insourcing or outsourcing). The minimum of the range is assumed to be zero with the exception of outsourcing variables on constraint 5 where a minimum of 0.39 on the coefficient range is defined.

Next, tightness (τ) of each constraint is defined. In Table A-1 the τ values are defined for each of the 5 machines. The same τ value is used for both the machine time and labor time constraints associated with each machine.

The next 4 columns in Table A-1 define the objective function coefficients. The mean and variance is defined for both the insourcing variable coefficients (p) and the outsourcing variable coefficients (c). This factor was held constant for all data sets

Finally, the columns labeled "Routing Sizes (%)" display the routing size for both the insourcing and outsourcing variables. The maximum routing size is the maximum percent of the parts with the same routing. It is defined in more detail in Section 6.2. The minimum routing size sets a minimum percent of parts that must visit the remaining routes (after the maximum route is defined). This effectively sets the the number of routings defined.

Table A-1: Industry-Motivated Problem Generation

			Maximum of Coefficient Range																			Routing Sizes (%)			
Set	#In Vars	#Out Vars	Insourcing Vars 1	2	3	4	5	Outsourcing Vars 1	2	3	4	5	τ 1	2	3	4	5	p μ	σ^2	c μ	σ^2	in max	min	out max	min
1	1600	400	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
2	1000	1000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
3	400	1600	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
4	1000	4000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
5	2500	2500	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
6	4000	1000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
7	1600	6400	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
8	4000	4000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
9	6400	1600	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
10	1600	400	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
11	1000	1000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
12	400	1600	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
13	1000	4000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
14	2500	2500	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
15	4000	1000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
16	1600	6400	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
17	4000	4000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
18	6400	1600	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
19	1600	400	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
20	1000	1000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
21	400	1600	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
22	1000	4000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
23	2500	2500	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
24	4000	1000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
25	1600	6400	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
26	4000	4000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
27	6400	1600	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
28	1600	400	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
29	1000	1000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
30	400	1600	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
31	1000	4000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
32	2500	2500	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
33	4000	1000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
34	1600	6400	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
35	4000	4000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
36	6400	1600	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
37	1600	400	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
38	1000	1000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20

continued on next page

Table A-1 continued from previous page

Set	#In Vars	#Out Vars	Maximum of Coefficient Range: Insourcing Vars 1	Insourcing Vars 2	Insourcing Vars 3	Insourcing Vars 4	Insourcing Vars 5	Outsourcing Vars 1	Outsourcing Vars 2	Outsourcing Vars 3	Outsourcing Vars 4	Outsourcing Vars 5	τ 1	τ 2	τ 3	τ 4	τ 5	p μ	p σ^2	c μ	c σ^2	Routing Sizes (%) in max	in min	out max	out min
39	400	1600	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
40	1000	4000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
41	2500	2500	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
42	4000	1000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
43	1600	6400	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
44	4000	4000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
45	6400	1600	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	30	20	30	20
46	1600	400	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
47	1000	1000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
48	400	1600	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
49	1000	4000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
50	2500	2500	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
51	4000	1000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
52	1600	6400	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
53	4000	4000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
54	6400	1600	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.30	0.09	0.17	0.24	0.31	20	25	20	25	80	2	80	2
55	1600	400	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
56	1000	1000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
57	400	1600	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
58	1000	4000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
59	2500	2500	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
60	4000	1000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
61	1600	6400	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
62	4000	4000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
63	6400	1600	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
64	1600	400	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
65	1000	1000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
66	400	1600	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
67	1000	4000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
68	2500	2500	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
69	4000	1000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
70	1600	6400	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
71	4000	4000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
72	6400	1600	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
73	1600	400	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
74	1000	1000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
75	400	1600	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
76	1000	4000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
77	2500	2500	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
78	4000	1000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
79	1600	6400	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
80	4000	4000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20

continued on next page

Table A-1 continued from previous page

Set	#In	#Out	Maximum of Coefficient Range																				Routing Sizes (%)			
			Insourcing Vars					Outsourcing Vars					τ					p		c		in		out	
	Vars	Vars	1	2	3	4	5	1	2	3	4	5	1	2	3	4	5	μ	σ^2	μ	σ^2	max	min	max	min
81	6400	1600	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
82	1600	400	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
83	1000	1000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
84	400	1600	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
85	1000	4000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
86	2500	2500	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
87	4000	1000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
88	1600	6400	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
89	4000	4000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
90	6400	1600	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
91	1600	400	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
92	1000	1000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
93	400	1600	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
94	1000	4000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
95	2500	2500	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
96	4000	1000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
97	1600	6400	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
98	4000	4000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
99	6400	1600	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	30	20	30	20
100	1600	400	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
101	1000	1000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
102	400	1600	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
103	1000	4000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
104	2500	2500	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
105	4000	1000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
106	1600	6400	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
107	4000	4000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
108	6400	1600	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.61	0.18	0.34	0.48	0.62	20	25	20	25	80	2	80	2
109	1600	400	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
110	1000	1000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
111	400	1600	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
112	1000	4000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
113	2500	2500	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
114	4000	1000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
115	1600	6400	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
116	4000	4000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
117	6400	1600	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
118	1600	400	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
119	1000	1000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
120	400	1600	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
121	1000	4000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
122	2500	2500	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2

continued on next page

Table A-1 continued from previous page

Set	#In Vars	#Out Vars	Maximum of Coefficient Range: Insourcing Vars 1	Insourcing Vars 2	Insourcing Vars 3	Insourcing Vars 4	Insourcing Vars 5	Outsourcing Vars 1	Outsourcing Vars 2	Outsourcing Vars 3	Outsourcing Vars 4	Outsourcing Vars 5	τ 1	τ 2	τ 3	τ 4	τ 5	p μ	p σ^2	c μ	c σ^2	Routing Sizes (%) in max	in min	out max	out min
123	4000	1000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
124	1600	6400	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
125	4000	4000	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
126	6400	1600	0.87	13.4	9.6	19.9	66.7	5.8	89.3	64.3	132.6	444.4	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
127	1600	400	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
128	1000	1000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
129	400	1600	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
130	1000	4000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
131	2500	2500	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
132	4000	1000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
133	1600	6400	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
134	4000	4000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
135	6400	1600	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
136	1600	400	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
137	1000	1000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
138	400	1600	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
139	1000	4000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
140	2500	2500	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
141	4000	1000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
142	1600	6400	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
143	4000	4000	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
144	6400	1600	8.69	134.0	96.4	198.9	666.6	11.6	178.6	128.5	265.2	888.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
145	1600	400	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
146	1000	1000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
147	400	1600	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
148	1000	4000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
149	2500	2500	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
150	4000	1000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
151	1600	6400	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
152	4000	4000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
153	6400	1600	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	30	20	30	20
154	1600	400	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
155	1000	1000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
156	400	1600	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
157	1000	4000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
158	2500	2500	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
159	4000	1000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
160	1600	6400	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
161	4000	4000	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2
162	6400	1600	86.93	1339.8	963.8	1988.7	6665.6	58.0	893.2	642.5	1325.8	4443.7	0.91	0.28	0.50	0.71	0.93	20	25	20	25	80	2	80	2

Table A-2 displays the results from experimental runs with the parameters listed in Table A-1. The results in each row represent the average of 10 problems generated using the parameters listed in that row. Tightness, Range of Coefficients, Maximum Block Size, Number of Parts, and Part Ratio are as described above for Table A-1. The column labeled % in-sourced is the percent of variables, out of the possible variables that can be insourced into the facility, insourced in the solution. The column labeled % out-sourced is the percent of variables in the current load that were outsourced in the solution. The row labeled Solution Time is the average CPU seconds that CPLEX requires to solve the problem optimally.

Table A-2: Results from Single-Facility Industry-Motivated Data Sets

Set	**Tightness**	**Range of Coefficients**	**Max Block Size**	**No. Parts**	**Part Ratio**	**% in-sourced**	**% out-sourced**	**Solution Time (CPU)**
	(1,2,3)	**(1,2,3)**	**(30,80)**	**(2,5,8K)**	**(0.25,1,4)**			
1	1	1	30	2	0.25	61%	4%	62.273
2	1	1	30	2	1	46%	4%	10.920
3	1	1	30	2	4	19%	2%	6.056
4	1	1	30	5	0.25	20%	2%	9.203
5	1	1	30	5	1	46%	4%	0.669
6	1	1	30	5	4	61%	4%	0.348
7	1	1	30	8	0.25	20%	2%	1.977
8	1	1	30	8	1	46%	4%	0.308
9	1	1	30	8	4	61%	4%	0.377
10	1	1	80	2	0.25	62%	4%	0.536
11	1	1	80	2	1	46%	3%	0.402
12	1	1	80	2	4	19%	1%	0.233
13	1	1	80	5	0.25	19%	1%	2.100
14	1	1	80	5	1	46%	3%	0.723
15	1	1	80	5	4	62%	4%	0.311
16	1	1	80	8	0.25	20%	1%	1.291
17	1	1	80	8	1	46%	3%	0.328
18	1	1	80	8	4	62%	4%	0.307
19	1	2	30	2	0.25	44%	9%	154.411
20	1	2	30	2	1	37%	13%	144.136
21	1	2	30	2	4	18%	7%	78.159
22	1	2	30	5	0.25	18%	7%	9.164
23	1	2	30	5	1	37%	13%	2.161
24	1	2	30	5	4	44%	9%	2.380
25	1	2	30	8	0.25	18%	8%	1.761

continued on next page

Table A-2 continued from previous page

Set	Tightness	Range of Coefficients	Max Block Size	No. Parts	Part Ratio	% in-sourced	% out-sourced	Solution Time (CPU)
26	1	2	30	8	1	37%	13%	0.330
27	1	2	30	8	4	44%	9%	0.430
28	1	2	80	2	0.25	47%	7%	1.078
29	1	2	80	2	1	37%	11%	0.680
30	1	2	80	2	4	18%	6%	0.580
31	1	2	80	5	0.25	18%	6%	0.690
32	1	2	80	5	1	38%	11%	0.481
33	1	2	80	5	4	47%	8%	0.608
34	1	2	80	8	0.25	18%	6%	1.085
35	1	2	80	8	1	38%	11%	0.277
36	1	2	80	8	4	47%	7%	0.321
37	1	3	30	2	0.25	39%	10%	74.664
38	1	3	30	2	1	32%	17%	354.953
39	1	3	30	2	4	17%	13%	316.169
40	1	3	30	5	0.25	17%	13%	5.020
41	1	3	30	5	1	32%	18%	1.770
42	1	3	30	5	4	39%	10%	1.067
43	1	3	30	8	0.25	17%	13%	0.799
44	1	3	30	8	1	32%	18%	0.436
45	1	3	30	8	4	39%	10%	0.422
46	1	3	80	2	0.25	43%	9%	0.885
47	1	3	80	2	1	33%	15%	0.536
48	1	3	80	2	4	17%	11%	0.375
49	1	3	80	5	0.25	17%	11%	0.892
50	1	3	80	5	1	33%	15%	0.650
51	1	3	80	5	4	43%	9%	0.584
52	1	3	80	8	0.25	17%	11%	0.438
53	1	3	80	8	1	33%	15%	0.325
54	1	3	80	8	4	43%	9%	0.511
55	2	1	30	2	0.25	68%	3%	4.700
56	2	1	30	2	1	47%	2%	2.598
57	2	1	30	2	4	20%	1%	3.211
58	2	1	30	5	0.25	20%	1%	12.247
59	2	1	30	5	1	47%	2%	0.242
60	2	1	30	5	4	68%	3%	0.201
61	2	1	30	8	0.25	20%	1%	1.294
62	2	1	30	8	1	47%	2%	0.264
63	2	1	30	8	4	68%	3%	0.449
64	2	1	80	2	0.25	70%	2%	0.267
65	2	1	80	2	1	47%	2%	1.066
66	2	1	80	2	4	20%	1%	0.149
67	2	1	80	5	0.25	20%	1%	0.683
68	2	1	80	5	1	47%	2%	0.756
69	2	1	80	5	4	70%	2%	0.394
70	2	1	80	8	0.25	20%	1%	1.290

continued on next page

Table A-2 continued from previous page

Set	Tightness	Range of Coefficients	Max Block Size	No. Parts	Part Ratio	% in-sourced	% out-sourced	Solution Time (CPU)
71	2	1	80	8	1	47%	2%	0.622
72	2	1	80	8	4	69%	2%	0.236
73	2	2	30	2	0.25	57%	5%	5.294
74	2	2	30	2	1	42%	8%	7.675
75	2	2	30	2	4	19%	4%	74.305
76	2	2	30	5	0.25	19%	5%	1.195
77	2	2	30	5	1	42%	8%	0.623
78	2	2	30	5	4	58%	5%	0.253
79	2	2	30	8	0.25	19%	5%	0.357
80	2	2	30	8	1	42%	8%	0.253
81	2	2	30	8	4	58%	5%	0.328
82	2	2	80	2	0.25	61%	4%	0.826
83	2	2	80	2	1	43%	6%	1.082
84	2	2	80	2	4	19%	3%	0.729
85	2	2	80	5	0.25	19%	4%	1.525
86	2	2	80	5	1	43%	6%	0.494
87	2	2	80	5	4	61%	4%	0.442
88	2	2	80	8	0.25	19%	4%	1.047
89	2	2	80	8	1	43%	6%	0.212
90	2	2	80	8	4	61%	4%	0.268
91	2	3	30	2	0.25	54%	6%	7.109
92	2	3	30	2	1	39%	11%	5.844
93	2	3	30	2	4	18%	8%	17.757
94	2	3	30	5	0.25	18%	8%	0.697
95	2	3	30	5	1	39%	11%	0.206
96	2	3	30	5	4	54%	6%	0.269
97	2	3	30	8	0.25	18%	8%	0.297
98	2	3	30	8	1	39%	11%	0.502
99	2	3	30	8	4	55%	6%	0.313
100	2	3	80	2	0.25	59%	5%	0.708
101	2	3	80	2	1	40%	8%	0.846
102	2	3	80	2	4	18%	6%	0.538
103	2	3	80	5	0.25	18%	6%	1.185
104	2	3	80	5	1	40%	8%	0.663
105	2	3	80	5	4	59%	5%	0.250
106	2	3	80	8	0.25	18%	6%	0.474
107	2	3	80	8	1	40%	8%	0.341
108	2	3	80	8	4	59%	5%	0.288
109	3	1	30	2	0.25	73%	2%	0.534
110	3	1	30	2	1	48%	1%	2.705
111	3	1	30	2	4	20%	1%	0.227
112	3	1	30	5	0.25	20%	1%	4.010
113	3	1	30	5	1	48%	2%	0.217
114	3	1	30	5	4	72%	2%	0.140
115	3	1	30	8	0.25	20%	1%	2.863

continued on next page

Table A-2 continued from previous page

Set	Tightness	Range of Coefficients	Max Block Size	No. Parts	Part Ratio	% in-sourced	% out-sourced	Solution Time (CPU)
116	3	1	30	8	1	48%	2%	0.212
117	3	1	30	8	4	72%	2%	0.175
118	3	1	80	2	0.25	76%	0%	0.881
119	3	1	80	2	1	49%	0%	0.245
120	3	1	80	2	4	20%	0%	0.163
121	3	1	80	5	0.25	20%	0%	0.553
122	3	1	80	5	1	49%	0%	0.614
123	3	1	80	5	4	76%	0%	0.512
124	3	1	80	8	0.25	20%	0%	0.706
125	3	1	80	8	1	49%	0%	0.637
126	3	1	80	8	4	76%	0%	0.225
127	3	2	30	2	0.25	64%	3%	1.078
128	3	2	30	2	1	44%	5%	0.981
129	3	2	30	2	4	19%	3%	0.773
130	3	2	30	5	0.25	19%	3%	1.559
131	3	2	30	5	1	44%	5%	0.137
132	3	2	30	5	4	65%	4%	0.136
133	3	2	30	8	0.25	19%	3%	0.486
134	3	2	30	8	1	44%	5%	0.174
135	3	2	30	8	4	65%	4%	0.169
136	3	2	80	2	0.25	74%	1%	0.821
137	3	2	80	2	1	47%	1%	1.102
138	3	2	80	2	4	19%	1%	0.230
139	3	2	80	5	0.25	19%	1%	1.236
140	3	2	80	5	1	47%	1%	0.886
141	3	2	80	5	4	75%	1%	0.323
142	3	2	80	8	0.25	19%	1%	1.659
143	3	2	80	8	1	47%	1%	0.184
144	3	2	80	8	4	74%	1%	0.369
145	3	3	30	2	0.25	63%	4%	0.389
146	3	3	30	2	1	42%	7%	1.439
147	3	3	30	2	4	19%	5%	29.744
148	3	3	30	5	0.25	19%	5%	1.055
149	3	3	30	5	1	42%	7%	0.136
150	3	3	30	5	4	62%	4%	0.139
151	3	3	30	8	0.25	19%	5%	0.182
152	3	3	30	8	1	42%	7%	0.174
153	3	3	30	8	4	62%	4%	0.183
154	3	3	80	2	0.25	74%	1%	1.176
155	3	3	80	2	1	47%	2%	1.326
156	3	3	80	2	4	19%	1%	0.495
157	3	3	80	5	0.25	19%	1%	1.509
158	3	3	80	5	1	47%	2%	0.491
159	3	3	80	5	4	74%	1%	0.130
160	3	3	80	8	0.25	19%	1%	0.904

continued on next page

Table A-2 continued from previous page								
Set	**Tightness**	**Range of Coefficients**	**Max Block Size**	**No. Parts**	**Part Ratio**	**% in-sourced**	**% out-sourced**	**Solution Time (CPU)**
161	3	3	80	8	1	47%	2%	0.175
162	3	3	80	8	4	74%	1%	0.144
					Average	41%	5%	9.205

The problems in Table A-3 are generated using the Freville and Plateau [12] method in which $p_i = \left(\sum_{j=1}^{m} A_{ij}\right)/m + Kr_i$, where A_{ij} are the constraint coefficients, p_i are the objective function coefficients, and r_i is a random number from $U(0,1)$. Additionally, $b_j = \tau \sum_{i=1}^{n} A_{ij}$, $\forall j = 1, \ldots, m$ and the A_{ij} values are generated from a uniform distribution on the defined range for each constraint.

Table A-3: Results with Single-Facility Standard Literature Generation Methods

Set	τ	**K**	**MC**	**Coefficient Ranges 1**	**2**	**3**	**4**	**5**	**Soln Time (CPU)**	**% In-sourced**	**% Out-sourced**
1	0.25	50	Min:	50	100	100	200	100	1205	61	36
2	0.50		Max:	100	150	200	250	250	1023	73	23
3	0.75								537	87	12
4	0.25	500	Min:	1	1000	3000	6000	10000	4308	65	40
5	0.50		Max:	1000	3000	6000	10000	17000	2993	74	24
6	0.75								1645	87	12
7	0.25	5000	Min:	1	10000	20000	50000	100000	3064	62	37
8	0.50		Max:	10000	20000	50000	100000	160000	2173	75	24
9	0.75								954	87	12
								Average	**1989**	**75%**	**24%**

In Table A-4, the results are presented from experimentation on the two-facility model with the industry-motivated data sets. The 54 different scenarios are the same scenarios solved for the single-plant problem with the exclusion of the unique part ratio parameter. With greater than one facility, the part ratio and the number of parts cannot easily be separated. In Table A-4 the average CPU seconds required to solve the problem optimally in CPLEX is displayed for each scenario.

Table A-4: Results from Two-Facility Experimentation

Set	Contstraint Tightness	Range of Coefficients	Max Route Size	Total No. Parts	Part Ratio	Solution Time (CPU)
	(1,2,3)	(1,2,3)	(30,80)	(4,10,16K)	(1,4)	
1	3	2	30	4000	1	0.22
2	2	2	30	4000	1	0.25
3	1	2	30	4000	1	0.31
4	3	1	30	4000	1	0.20
5	2	1	30	4000	1	0.23
6	1	1	30	4000	1	0.24
7	3	3	30	4000	1	1.52
8	2	3	30	4000	1	15.86
9	1	3	30	4000	1	3732.53
10	3	2	80	4000	1	2683.31
11	2	2	80	4000	1	0.52
12	1	2	80	4000	1	0.83
13	3	1	80	4000	1	0.34
14	2	1	80	4000	1	0.20
15	1	1	80	4000	1	0.22
16	3	3	80	4000	1	4.61
17	2	3	80	4000	1	78.72
18	1	3	80	4000	1	175.09
19	3	2	30	4000	1	6.10
20	2	2	30	16000	1	1.46
21	1	2	30	16000	1	1.65
22	3	1	30	16000	1	1.04
23	2	1	30	16000	1	1.11
24	1	1	30	16000	1	1.34
25	3	3	30	16000	1	2.01
26	2	3	30	16000	1	2.71
27	1	3	30	16000	1	209.94
28	3	2	80	16000	1	1.82
29	2	2	80	16000	1	1.34
30	1	2	80	16000	1	13.42
31	3	1	80	16000	1	6.69
32	2	1	80	16000	1	0.92
33	1	1	80	16000	1	1.16
34	3	3	80	16000	1	2.53
35	2	3	80	16000	1	16.13
36	1	3	80	16000	1	165.74
37	3	2	30	16000	1	17.01
38	2	2	30	10000	4	57.23
39	1	2	30	10000	4	8189.46
40	3	1	30	10000	4	4.95

continued on next page

Table A-4 continued from previous page						
Set	**Contstraint Tightness**	**Range of Coefficients**	**Max Route Size**	**Total No. Parts**	**Part Ratio**	**Solution Time (CPU)**
41	2	1	30	10000	4	254.69
42	1	1	30	10000	4	4835.30
43	3	3	30	10000	4	1.43
44	2	3	30	10000	4	50.55
45	1	3	30	10000	4	11243.26
46	3	2	80	10000	4	6.36
47	2	2	80	10000	4	68.73
48	1	2	80	10000	4	5984.84
49	3	1	80	10000	4	18.45
50	2	1	80	10000	4	8782.85
51	1	1	80	10000	4	7692.54
52	3	3	80	10000	4	9.59
53	2	3	80	10000	4	826.50
54	1	3	80	10000	4	15607.58
					average:	**1310.81**

Table A-5 displays the results from solving the same set of problems as in Table A-4, but with the heuristic presented in Chapter 5. The CPU seconds required for both the heuristic solution and the optimal solution are displayed. The final column is the gap between the optimal and heuristic solution values.

Table A-5: Results from Two-Facility Heuristic Experimentation

Set	**Coeff. Range**	**Tight- ness**	**Max Route**	**Tot No. Parts**	**Part Ratio**	**Heuristic Solu Time**	**Optimal Solu Time**	**Heuristic Solu Gap**
	(1,2,3)	**(1,2,3)**	**(30,80)**	**(4,10,16K)**	**(1, 4)**	**(CPU sec.)**	**(CPU sec.)**	
1	2	3	30	4K	1	0.12	0.22	0.00%
2	2	2	30	4K	1	0.13	0.25	0.00%
3	2	1	30	4K	1	0.12	0.31	0.00%
4	1	3	30	4K	1	0.11	0.20	0.00%
5	1	2	30	4K	1	0.11	0.23	0.00%
6	1	1	30	4K	1	0.11	0.24	0.00%
7	3	3	30	4K	1	0.16	1.52	0.08%
8	3	2	30	4K	1	0.16	15.86	0.04%
9	3	1	30	4K	1	0.17	3732.53	0.22%
10	2	3	80	4K	1	0.12	2683.31	0.00%
11	2	2	80	4K	1	0.16	0.52	0.09%
12	2	1	80	4K	1	0.16	0.83	0.16%
13	1	3	80	4K	1	0.11	0.34	0.00%

continued on next page

Table A-5 continued from previous page								
Set	**Coeff. Range**	**Tight-ness**	**Max Route**	**Tot No. Parts**	**Part Ratio**	**Heuristic Solu Time**	**Optimal Solu Time**	**Heuristic Solu Gap**
14	1	2	80	4K	1	0.11	0.20	0.00%
15	1	1	80	4K	1	0.11	0.22	0.00%
16	3	3	80	4K	1	0.16	4.61	0.07%
17	3	2	80	4K	1	0.16	78.72	0.10%
18	3	1	80	4K	1	0.16	175.09	0.15%
19	2	3	30	16K	1	1.23	7.09	0.00%
20	2	2	30	16K	1	1.48	1.45	0.00%
21	2	1	30	16K	1	1.48	1.70	0.00%
22	1	3	30	16K	1	1.40	1.06	0.00%
23	1	2	30	16K	1	1.41	1.11	0.00%
24	1	1	30	16K	1	1.32	1.34	0.00%
25	3	3	30	16K	1	1.58	2.01	0.00%
26	3	2	30	16K	1	1.58	2.71	0.01%
27	3	1	30	16K	1	1.62	209.94	0.05%
28	2	3	80	16K	1	1.61	1.82	0.00%
29	2	2	80	16K	1	1.69	1.34	0.01%
30	2	1	80	16K	1	1.68	13.42	0.05%
31	1	3	80	16K	1	1.28	6.69	0.00%
32	1	2	80	16K	1	1.31	0.92	0.00%
33	1	1	80	16K	1	1.31	1.16	0.00%
34	3	3	80	16K	1	1.64	2.53	0.01%
35	3	2	80	16K	1	1.89	16.13	0.02%
36	3	1	80	16K	1	1.64	165.74	0.05%
37	2	3	30	16K	1	0.99	17.01	0.02%
38	2	2	30	10K	4	0.71	57.23	0.05%
39	2	1	30	10K	4	0.78	8189.46	0.14%
40	1	3	30	10K	4	0.71	4.95	0.08%
41	1	2	30	10K	4	0.74	254.69	0.20%
42	1	1	30	10K	4	0.76	4835.30	0.34%
43	3	3	30	10K	4	0.65	1.43	0.02%
44	3	2	30	10K	4	0.68	50.55	0.08%
45	3	1	30	10K	4	0.88	11243.26	0.09%
46	2	3	80	10K	4	0.65	6.36	0.04%
47	2	2	80	10K	4	0.70	68.73	0.06%
48	2	1	80	10K	4	0.74	5984.84	0.10%
49	1	3	80	10K	4	0.71	18.45	0.15%
50	1	2	80	10K	4	0.74	8782.85	0.20%
51	1	1	80	10K	4	0.91	7692.54	0.26%
52	3	3	80	10K	4	0.63	9.59	0.03%
53	3	2	80	10K	4	0.66	826.50	0.08%
54	3	1	80	10K	4	0.79	15607.58	0.08%
					Average	**0.8**	**1310.8**	**0.06%**

Table A-6 presents the results from experimentation with three facilities. Each of the three facilities in a problem are generated with identical parameters. Additionally, the coefficients for each facility are generated with the same parameters as in the single-facility problems in Tables A-4 and A-5. The number of parts that can be outsourced from each facility remains the same; however, the number of parts that can be insourced increases because the additional facility outsources parts, and thus brings additional parts that can be insourced to the other two facilities. In Table A-6 the problem parameters and the CPU time required to solve the problem optimally in CPLEX is displayed.

Table A-6: Results from Three-Facility Experimentation

Set	**Contstraint Tightness**	**Range of Coefficients**	**Max Route Size**	**Total No. Parts**	**Solution Time (CPU)**
	(1,2,3)	**(1,2,3)**	**(30,80)**	**(9,12,18K)**	
1	3	2	30	9K	0.44
2	2	2	30	9K	0.79
3	1	2	30	9K	2.07
4	3	1	30	9K	0.51
5	2	1	30	9K	1.56
6	1	1	30	9K	3.52
7	3	3	30	9K	0.68
8	2	3	30	9K	5.92
9	1	3	30	9K	1389.27
10	3	2	80	9K	53.16
11	2	2	80	9K	0.91
12	1	2	80	9K	4.90
13	3	1	80	9K	2.70
14	2	1	80	9K	0.94
15	1	1	80	9K	4.78
16	3	3	80	9K	3.36
17	2	3	80	9K	6.74
18	1	3	80	9K	32.44
19	3	2	30	12K	7.12
20	2	2	30	12K	0.96
21	1	2	30	12K	18.25
22	3	1	30	12K	6.61
23	2	1	30	12K	5.85
24	1	1	30	12K	10.16
25	3	3	30	12K	2.55
26	2	3	30	12K	15.47
27	1	3	30	12K	294.80
28	3	2	80	12K	55.50

continued on next page

Table A-6 continued from previous page					
Set	**Contstraint Tightness**	**Range of Coefficients**	**Max Route Size**	**Total No. Parts**	**Solution Time (CPU)**
29	2	2	80	12K	0.78
30	1	2	80	12K	10.28
31	3	1	80	12K	3.17
32	2	1	80	12K	1.88
33	1	1	80	12K	20.21
34	3	3	80	12K	3.16
35	2	3	80	12K	1.53
36	1	3	80	12K	12.45
37	3	2	30	18K	20.95
38	2	2	30	18K	1.17
39	1	2	30	18K	2.23
40	3	1	30	18K	1.69
41	2	1	30	18K	1.14
42	1	1	30	18K	1.85
43	3	3	30	18K	2.11
44	2	3	30	18K	2.01
45	1	3	30	18K	86.50
46	3	2	80	18K	148.11
47	2	2	80	18K	1.13
48	1	2	80	18K	11.66
49	3	1	80	18K	1.38
50	2	1	80	18K	1.75
51	1	1	80	18K	5.69
52	3	3	80	18K	0.94
53	2	3	80	18K	2.01
54	1	3	80	18K	59.74
				Average	**43.29**

Table A-7 displays the results from experimentation with four facilities. The four facilities per problem are generated with identical parameters. Additionally, the coefficients for each facility are generated with the same parameters as in the single-facility problems in Tables A-4, A-5, and A-6. As with three facilities, the number of parts that can be insourced into each facility again increases with the addition of the fourth facility. In Table A-7 the problem parameters and the CPU time required to solve the problem optimally in CPLEX is shown.

Table A-7: Results from Three-Facility Experimentation

Set	**Contstraint Tightness**	**Range of Coefficients**	**Max Route Size**	**Total No. Parts**	**Solution Time (CPU)**
	(1,2,3)	**(1,2,3)**	**(30,80)**	**(9,12,18K)**	
1	3	2	30	12K	0.78
2	2	2	30	12K	0.96
3	1	2	30	12K	18.98
4	3	1	30	12K	0.96
5	2	1	30	12K	3.90
6	1	1	30	12K	1.89
7	3	3	30	12K	0.83
8	2	3	30	12K	2.14
9	1	3	30	12K	109.86
10	3	2	80	12K	24.43
11	2	2	80	12K	1.74
12	1	2	80	12K	8.56
13	3	1	80	12K	5.90
14	2	1	80	12K	3.38
15	1	1	80	12K	4.42
16	3	3	80	12K	3.26
17	2	3	80	12K	19.90
18	1	3	80	12K	42.31
19	3	2	30	20K	8.95
20	2	2	30	20K	1.27
21	1	2	30	20K	12.47
22	3	1	30	20K	1.32
23	2	1	30	20K	1.49
24	1	1	30	20K	20.43
25	3	3	30	20K	10.40
26	2	3	30	20K	1.92
27	1	3	30	20K	48.93
28	3	2	80	20K	46.03
29	2	2	80	20K	1.39
30	1	2	80	20K	26.83
31	3	1	80	20K	31.29
32	2	1	80	20K	12.11
33	1	1	80	20K	28.70
34	3	3	80	20K	6.80
35	2	3	80	20K	1.59
36	1	3	80	20K	37.64
37	3	2	30	32K	44.67
38	2	2	30	32K	2.23
39	1	2	30	32K	4.01
40	3	1	30	32K	2.53

continued on next page

Table A-7 continued from previous page

Set	**Contstraint Tightness**	**Range of Coefficients**	**Max Route Size**	**Total No. Parts**	**Solution Time (CPU)**
41	2	1	30	32K	2.24
42	1	1	30	32K	3.13
43	3	3	30	32K	5.41
44	2	3	30	32K	3.15
45	1	3	30	32K	31.74
46	3	2	80	32K	7.06
47	2	2	80	32K	1.60
48	1	2	80	32K	3.05
49	3	1	80	32K	3.26
50	2	1	80	32K	1.82
51	1	1	80	32K	2.72
52	3	3	80	32K	26.12
53	2	3	80	32K	2.23
54	1	3	80	32K	16.03
				Average	**13.27**

Vita

Natalie S. Cherbaka received her B.S. in Mathematics from Taylor University and her M.S. in Industrial Engineering from North Carolina State University. Her professional experience is as a fulfillment manager for the IBM Personal Systems Division. In 2001 she commenced her Ph.D. studies at Virginia Tech in the Grado Department of Industrial & Systems Engineering and completed this work in December 2004. While working towards the Ph.D. degree, she received the *AGVS & ISC Product Section Honor Scholarship* from the Material Handling Education Foundation. She is a student member of IIE, INFORMS, and Alpha Pi Mu.