STRATEGIC SURVEILLANCE SYSTEM DESIGN

FOR PORTS AND WATERWAYS

A Dissertation

by

ELİF İLKE ÇİMREN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

May 2009

Major Subject: Industrial Engineering

STRATEGIC SURVEILLANCE SYSTEM DESIGN

FOR PORTS AND WATERWAYS

A Dissertation

by

ELİF İLKE ÇİMREN

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Chair of Committee,	Wilbert E. Wilhelm
Committee Members,	Donald K. Friesen
	Sergiy Butenko
	Kiavash Kianfar
Head of Department,	Brett A. Peters

May 2009

Major Subject: Industrial Engineering

ABSTRACT

Strategic Surveillance System Design for Ports and Waterways. (May 2009)

Elif İlke Çimren, B.S., Istanbul Technical University;

M.S., Sabanci University

Chair of Advisory Committee: Dr. Wilbert E. Wilhelm

The purpose of this dissertation is to synthesize a methodology to prescribe a strategic design of a surveillance system to provide the required level of surveillance for ports and waterways. The method of approach to this problem is to formulate a linear integer programming model to prescribe a strategic surveillance system design (SSD) for ports or waterways, to devise branch-and-price decomposition (B&P-D) and branch-and-cut (B&C) methodologies to solve real-size (i.e., large-scale) SSD problems (SSDPs), and to compare the efficacies of B&P-D and B&C procedures.

The first part of this dissertation formulates SSDP as an integer programming model. The model represents relevant practical considerations and prescribes the types of sensors, the number of each type, and the location of each sensor to meet surveillance requirements while minimizing total cost. The resulting model is a multidimensional knapsack problem with generalized upper bound constraints (GUBs).

The second part of this dissertation designs a B&P-D to solve SSDP. We evaluate alternative ways of formulating and implementing B&P-D and identify default B&P-D, which requires less run time than the others. We use data representing the

Houston Ship Channel as a test bed to evaluate the efficacy of the default B&P-D, benchmarking relative to a commercial solver and analyzing the influence of parameters (i.e., experimantal factors) on run time. Our results show that the default B&P-D requires less run time than CPLEX B&B and provides strong bounds. Tests also show that the run time of B&P-D increases with the number of GUBs.

The third part of this dissertation characterizes a family of valid inequalities - α-*cover* inequalities - for the knapsack polytope with GUBs (KPG) along with a procedure to generate them. It presents necessary and sufficient conditions under which these inequalities are facets of KPG polytope, and demonstrates how they can be lifted otherwise. Furthermore, it devises a separation procedure to cut off a fractional solution to the linear relaxation of KPG and presents computational results to evaluate the efficacy of the α-cover cuts. Computational tests show that α-cover cuts provide tighter cuts than either surrogate-knapsack or lifted cover cuts and using them to generate cuts for 0-1 integer problems with multiple constraints requires less run time.

In the last part of the dissertation, using SSDP instances of real size and scope, we compare the efficacy of B&C, which uses α - *cover* inequalities as cuts, and B&P-D approaches. Our results show the B&C method, which detects a violated α-cover inequality for each knapsack and adds it after modifying it by lifting to be a facial inequality, is the fastest of the methods. We also analyze the sensitivity of the system and the cost to important parameters. The sensitivity analysis shows that cost is relatively insensitive to changes in parameters.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor, Professor Wilbert E. Wilhelm, for introducing the strategic surveillance system design problem to me; for his constant support; and also for his patience teaching me how to write a paper. Throughout my Ph.D. studies, he has not only guided or contributed to my life educationally, but also taught me so many aspects of life. Without his support I could not have completed my Ph.D. study smoothly.

This dissertation contains three working papers co-authored with Professor Wilhelm. These working papers are under review for publication. One of these working papers is accepted by *IEEE Transactions on Automation Science and Engineering*, but the others have not been accepted yet. I also would like to thank Professor Wilhelm for his helpful comments and suggestions that have improved both the content and the structure of these papers. Much of the wording in this dissertation was taken from these papers and was composed or edited by Professor Wilhelm.

I owe a special acknowledgement to Major Patrick Walden for his patience and time answering my endless questions about sensors and security strategies. I wish to thank Professor Donald K. Friesen, Dr. Sergiy Butenko, and Dr. Kiavash Kianfar for serving on the committee and for giving helpful suggestions. Further, thanks to Dr. Yu Ding, Dr. Jung Jin Cho, and Abhishek K. Shrivastava for the valuable discussions and the exchange of information.

TABLE OF CONTENTS

CHAPTER Page

LIST OF FIGURES

LIST OF TABLES

CHAPTER I

INTRODUCTION

Strategic surveillance system design (SSD) prescribes the type of sensors, the number of each type, and the location of each sensor to achieve the required level of surveillance. This dissertation synthesizes a methodology to prescribe a SSD to provide the required level of surveillance for ports and waterways. It fulfills its purpose in three related parts:

(i) formulation of the strategic SSD problem (SSDP) for ports and waterways;

(ii) a branch-and-price (B&P) decomposition (B&P-D) approach, including evaluation of alternative B&P-Ds of the multidimensional knapsack problem with generalized upper bound (GUB) constraints (MKGP) with the goal of establishing relationships among the bounds these methods provide - both analytically and computationally;

(iii) a branch-and-cut (B&C) approach to solve large-scale SSDPs.

In the first part of this dissertation we formulate a linear integer programming model to prescribe a SSD for a port or waterway. The resulting model is in the form of a MKGP. In previous studies, B&P and B&C procedures have been used successfully for solving 0-1 integer problems. Motivated by this, the second and third parts of this dissertation are, respectively, B&P-D and B&C solution procedures to solve the SSDP.

In the second part, we explore various B&P-Ds that might be applied to MKGP

This dissertation follows the style of *Operations Research*.

with the goal of identifying an effective means of implementing B&P-D for solving SSDP exactly. As part of our theoretical analysis we compare the bounds available from B&P-Ds with two alternative relaxations (Lagrangian relaxation, Lagrangian decomposition) and determine whether incorporating a surrogate constraint can make an improvement or not. Our computational tests compare alternative ways of implementing B&P-D to assess the trade-off between the tightness of resulting bounds and the run times required to obtain them. Then, we use the B&P-D formulation that requires less run time than the others to solve SSDP instances.

In the third part, we identify valid inequalities (facets) for knapsack problem with GUBs (KGP), which is a subproblem of MKGP. Then, we use these cuts to solve SSDP instances of realistic size by B&C.

In this chapter we give a brief overview of the research. Section 1.1 reviews literature on SSDP, addressing issues important to SSD. Section 1.2 presents our research motivation. Section 1.3 specifies our research objectives. Finally, Section 1.4 concludes this chapter by presenting the organization of the dissertation.

1.1. Background

Sensor location problems have received considerable attention recently. The typical research paper focuses on a specific application such as the control of distributed process systems (e.g., chemical reactors) (Alonso et al. 2004, Bagajewicz and Cabrere 2002), parameter estimation in structural dynamics (Papadimitriou 2005), or contaminant detection in municipal water networks (Berry et al. 2004). A number of works have also dealt with selecting the minimal number of sensors to maintain coverage and connectivity in a network (Akyildiz et al. 2002, Zou and Chakrabarty 2005). Another variant of the problem involves network interdiction, for example, locating sensors to minimize the probability that a smuggler can travel through a transportation network undetected (Morton et al. 2007).

Relatively little research has focused on surveillance, especially for the port-and-waterway environment. Prior work for port-and-waterway surveillance is, by and large, qualitative (Kharchenko and Vasylyev 2002). In a recent exception, Ben-Zvi and Nickerson (2007) presented an algorithm to locate sensors that detect underwater threats; however, it considered a limited set of characteristics of intruders.

The density of wireless sensors airdropped in an area may be important in applications like seismic analysis and environmental monitoring for agriculture (Mainwaring et al. 2002), but exact locations are important in surveillance applications (Clark 2004). One method (Fernandes et al. 2006) prescribes the locations for a predetermined number of Light Detection and Ranging (LIDAR) stations to maximize surveillance coverage of a specified area. Visibility is based on the fact that LIDAR detects smoke plumes. Another algorithm (Pandit and Ferreira 1992) uses a set covering model, which is NP-hard, to prescribe a minimal number of sensors to provide surveillance of the edges of all objects (polygons) in an area. Sensor-location algorithms have been devised to assure observation of the entire surface of 2D (i.e., an area) (Bottino and Laurentini 2004) and 3D (i.e., a volume) (Bottino and Laurentini 2005) objects with the minimum number of sensors. These algorithms (Bottino and Laurentini

2004, 2005) were based on the assumption that a given “area” comprises simple polygons, but ports and waterways take on irregular shapes.

Methods are available to design a surveillance system by locating sensors on a grid. One approach (Chakrabarty et al. 2002) formulated a linear integer program to minimize the cost of locating sensors with different ranges to cover all grid points, each by at least a specified number of sensors. The authors proposed a theoretical framework and a divide-and-conquer approach to determine the best placement of sensors. Again employing a grid, Lin and Chiu (2005) formulated a combinatorial optimization model that minimizes the maximum sensor-to-surveillance-point distance under constraints that limit total cost and assure complete coverage. The authors devised a simulated annealing approach to solve the problem. Both Chakrabarty et al. (2002) and Lin and Chiu (2005) assumed that coverage is complete if the distance between a grid (surveillance) point and the sensor was within the detection range of the sensor; they ignored the geographical features of the corresponding area and did not consider sensor characteristics other than range. Furthermore, these papers employed the simplistic criterion that a surveillance point is covered if it is observed by a specified number of sensors. In another study, Kim and Park (2006) assumed that sensor capabilities decrease with distance, but ignored the effects of environmental conditions on sensor coverage and range. In contrast to Chakrabarty et al. (2002), Lin and Chiu (2005), and Kim and Park (2006), Park et al. (2004) focused on covering salient geographical features - such as roads, rivers, and buildings - rather than seeking complete coverage of the region.

Researchers have devised several approximate methods and heuristics. One polynomial-time approximation algorithm (Wang and Zhong 2006) seeks a minimum-cost sensor placement on a bounded, 3D field, which comprises a number of discrete points that may or may not be grid points. The model deals with different sensor types characterized by their ranges and costs, and every point in the field must be observed by at least a specified number of sensors. This algorithm first solves the linear relaxation of the problem using standard techniques and then converts a fractional solution to an integer solution in $O(n \log n)$ time. A heuristic procedure (Cavalier et al. 2007), based on Voronoi polygons, seeks to locate a finite number of identical sensors to detect an event in a given planar region, assumed to be a convex polygon. The objective is to minimize the maximum probability of non-detection.

Prior work has focused on locating sensors to observe a plane or grid, while ports and waterways take on irregular shapes, perhaps including long, narrow, and meandering paths. To our knowledge none of the prior work has considered the set of practicalities important in designing a surveillance system for port and waterway security:

(1) irregular shapes of ports and waterways;

(2) surveillance requirements; and

(3) the capabilities of each sensor type (e.g., radar; electro optical; infrared camera; seismic; electromagnetic; laser; sonar; and heat, motion, and radioactivity detectors), which may depend upon the time of day (e.g., lighting during morning, day, or night), weather conditions (e.g., rain, fog, snow, bright sun), unobstructed line of sight, and distance to a surveillance point.

The scope of this dissertation is surface surveillance and it focuses on filling these gaps left by earlier research. Although the overall methodology in this study is designed for solving the SSD for a ports or waterway, we expect that it could be adopted to deal with other applications like border patrol and underwater surveillance.

1.2. Motivation

Ports are installations where vessels can be loaded and unloaded, in particular, allowing passengers and cargo to enter a country through customs inspection. Specific examples of ports are Boston MA, New York NY, Miami FL, Houston TX, San Diego CA, San Francisco CA, and Seattle WA. A waterway is a navigable body of water, including rivers, bays, and channels. Examples are the Great Lakes, the Panama Canal, and the Ohio and Mississippi Rivers. Currently, the security of ports and waterways is the responsibility of the U.S. Coast Guard (USCG).

Immediately after the events of 9/11, the United States become aware of the destruction that a terrorist attack can cause and the urgency to prevent any reoccurrence of such an event. Each year, a huge number of ships that could carry destructive devices pass through U.S. ports; and a number of industries, which store and process both hazardous and flammable materials, line the shores of U.S. waterways. For example, according to Port of Houston website (2008), the Houston Ship Channel (HSC) daily imports over 11,000,000 barrels of petroleum and petroleum products (worth nearly $10 billion) and annually handles 1,000,000 containers. Moreover, a $15 billion petrochemical center that includes some of the world's largest plants lines its shore, and it is very close to populous areas.

The 2002 Maritime Transportation Security Act (Maritime Transportation Security Act 2008) requires that each large commercial cargo and passenger vessel install an automatic identification system (AIS) to provide detailed information about its identity to USCG Marine Safety Units (MSU). Hence, MSU knows the destinations of large vessels and can monitor them. On the other hand, small vessels, including barges and towing, fishing, and private recreation boats do not have such a requirement and none of them install AIS. Therefore, it is difficult to determine their intentions and monitor them. Due to their sizes, they can easily access critical regions, entering through a bayou that feeds the channel, launching at numerous locations along the channel or hiding in the shadow of a large vessel. Also, it is important to note that, since small boats usually can travel much faster than large vessels, early detection of a suspicious boat is important. Thus, critical (sensitive) regions along U.S. ports and waterways are threatened by intruders who can enter using small boats. The terrorist attack on the U.S.S. Cole on October 12, 2000 is evidence of the threat that a small vessel can cause (Congressional Research Service Report 2008).

Historically, USCG MSU has used television cameras and radars to monitor the ship channel, primarily to manage the flow of vessels. However, the current system is not enough to provide timely response to security threats posed, for example, by unauthorized small vessels. Because of the importance of this problem, USCG is interested in developing sensor surveillance systems to assure homeland security in U.S. ports and waterways. However, since resources are limited, USCG requires the design of cost-effective surveillance systems.

1.3. Research objectives

This dissertation has five research objectives; achieving them will fulfill the purpose of this study. The first objective is a linear integer programming model to prescribe a strategic design capable of providing an acceptable level of surveillance for a port or waterway. It is important that this model represent practical considerations important to port and waterway security. The second objective is an effective B&P-D approach to solve SSDP. Specifically, we explore several B&P-Ds formulations that might be applied to the MKGP, establishing relationships among the bounds these methods provide - both analytically and computationally. The third objective is a set of valid inequalities (facets) for the knapsack problem with GUBs (KGP). Then, we use these inequalities to solve SSDP by B&C. The fourth objective is a computational evaluation of B&P-D and B&C approaches and a comparison of them. The fifth objective is computational experience in solving SSD instances of realistic size and scope. For this purpose, we will use HSC, which is the sixth largest port in the world, as a test bed. It represents ports and waterways in general and its proximity allows us to gather information easily.

1.4. Organization of the dissertation

This dissertation is organized in eight chapters. Chapter II reviews literature relevant to this research. Chapter III formulates a linear integer programming model to prescribe a SSD for a port or waterway, addressing the first objective. Chapter IV identifies an effective B&P-D formulation (i.e., approach) to solve SSDP, addressing the second objective. Chapter V evaluates the B&P-D formulation that is identified in Chapter IV in an application that involves designing a surveillance system for port and waterway security, addressing the fourth objective. Chapter VI devises set of valid inequalities (facets) for KGP, addressing the third objective. Chapter VII uses the cuts generated in Chapter VI to solve SSDP by B&C and compares B&C approach with B&P-D, addressing the fourth and fifth objectives, respectively. In Chapter VIII we present our conclusions and some recommendations for future research.

CHAPTER II

LITERATURE REVIEW

This chapter reviews the literature related to this research. Since the integer programming formulation of the SSDP results in the form of MKGP, this chapter provides a review of MKGP. Section 2.1 presents the existing solution procedures for MKGP. Section 2.2 introduces problems related to MKGP and their solution methodologies with a detailed review of multiple-choice multidimensional knapsack problem (MCMKP) (since MKGP can be transformed to an MCMKP equivalently). Section 2.3 states the known relationships between the bounds provided by Lagrangian relaxation, surrogate and composite relaxations, and Lagrangian decomposition for integer programming problems (IPs). Section 2.4 reviews the literature on the KGP polytope and others related to it. Finally, Section 2.5 summarizes this entire chapter, emphasizing the necessity of this dissertation research.

2.1. Multidimensional knapsack problem with GUB constraints

In this dissertation, we consider the MKGP that is given in the following form:

$$\min\left\{cx : \sum_{g\in G}\sum_{j\in J_g} a_{ij}x_j \geq b_i \;\; i\in I; \sum_{j\in J_g} x_j \leq 1 \;\; g\in G; x_j \in \{0,1\} \;\; g\in G,\; j\in J_g\right\}.$$

MKGP is known to have a number of important applications, including underwater threat detection (Ben-Zvi and Nickerson 2007), sensor location (Kim and Park 2006), asset allocation (Li et al. 2004), and strategic surveillance system design (Section 3.1). Problems that contain multiple knapsack constraints are NP-hard in the strong sense

(Martello and Toth 1990), as is MKGP. To our knowledge, only heuristics have been proposed for MKGP (Li et al. 2004, Li and Curry 2005, Li 2005). It is important to note that the MKGP considered in the literature has knapsack constraints in the form of a less-than-equal-to inequalities $\left(\text{i.e.,} \sum_{g\in G}\sum_{j\in J_g} a_{ij}x_j \le b_i\right)$. Although polytopes associated with knapsacks in the form of less-than-or-equal-to and greater-than-or-equal-to inequalities are different from each other, they can be transformed to equivalent forms. Therefore, solution procedures for one of the forms can be used to solve the other form.

2.2. Variants of MKGP

MKGP is closely related to four other variants of the problem: 0-1 knapsack (KP), multiple-choice knapsack (MCKP), multidimensional knapsack (MKP) and MCMKP. Each of the following subsections reviews the literature on these problems.

2.2.1. Knapsack problem

KP, which is given by

$$\min\left\{cx : \sum_{g\in G}\sum_{j\in J_g} a_j x_j \le b;\, x_j \in \{0,1\}\; g \in G,\; j \in J_g\right\},$$

is a special case of MKGP. KP can be recast as MKGP by complementing each variable and forming n $\left(n = \sum_{g\in G}\left|J_g\right|\right)$ GUBs by assigning exactly one variable to each GUB (i.e., $x_j \le 1$). It is well known that KP is NP-hard (Garey and Johnson 1979). However, since it is not strongly NP-hard, it can be solved in pseudo-polynomial time by dynamic programming (Dantzig 1957, Martello and Toth 1990). Pseudo-polynomial algorithms, fully polynomial approximation schemes, search tree procedures and heuristics have also been proposed to solve KP (Kellerer et al. 2004, Martello et al. 1999).

2.2.2. Multiple-choice knapsack problem

MCKP, which is given by

$$\min\left\{cx : \sum_{g\in G}\sum_{j\in J_g} a_j x_j \le b \,;\, \sum_{j\in J_g} x_j = 1 \; g\in G; x_j \in \{0,1\} \; g\in G,\ j\in J_g\right\},$$

is a variation of KP in which variables are partitioned into classes and exactly one variable from each class must be set to 1. MCKP can be transformed into an equivalent MKGP by setting $\bar{c}_j = \max_{j'\in J_g} c_{j'} - c_j$, $\bar{a}_j = \max_{j'\in J_g} a_{j'} - a_j$ for $g \in G$, $j \in J_g$, and $\bar{b}_1 = \sum_{g\in G} \max_{j\in J_g} a_j - b$, and by eliminating one of the variables with $\bar{a}_j = 0$ from each class J_g in order to transform the multiple-choice equality into an inequality (Kellerer et al. 2004). MCKP was first introduced by Healy (1964) and in 1987 Dudzinski and Walukiewicz showed that it can be solved in pseudo-polynomial time. Since then many studies have dealt with it (Kellerer et al. 2004, Martello and Toth 1990, Pisinger 1995). Most of the algorithms for the exact solution of MCKP use B&B (Nauss 1978, Armstrong et al. 1983).

2.2.3. Multidimensional knapsack problem

MKP, which was introduced by Lorie and Savage (1955), involves multiple knapsack constraints, but no non-trivial GUBs. It is encountered in capital budgeting (Manne and Markowitz 1957), project selection (Petersen 1967), cutting stock (Gilmore and Gomory 1966) and loading problems (Shih 1979).

Since MKP is a well known to be NP-hard in the strong sense, finding a fully polynomial approximation algorithm is NP-hard (Magazine and Chern 1984). Hence, a number of studies have focused on preprocessing (Fréville and Plateau 1994), greedy heuristics (Toyoda 1975), metaheuristics (Chu and Beasley 1998, Hanafi and Fréville 1998), and approximate dynamic programming algorithms (Bertsimas and Demir 2002). A few exact algorithms are available to optimize MKP. They are based on dynamic programming (Gilmore and Gomory 1966, Weingartner and Ness 1967), branch-and-bound (B&B) (Shih 1979, Geoffrion 1974), hybrid algorithms combining dynamic programming and B&B (Marsten and Morin 1977), and implicit enumeration (Soyster et al. 1978). However, none solve MKP effectively and their applicability is typically limited to instances with relatively few variables and constraints. Moreover, dynamic programming can only be used to solve MKPs with small values of b_i (Fréville 2004). We refer the reader to Fréville (2004), Hanafi et al. (1996), Kellerer et al. (2004), Lin (1998), Stefan et al. (2008) for detailed information on solution approaches to MKP.

2.2.4. Multidimensional multiple-choice knapsack problem

As described by Moser et al. (1997), MCMKP has multiple knapsack constraints. MKGP can be transformed into an equivalent MCMKP (Kellerer et al. 2004) and the inverse is also true. To our knowledge, very few studies have focused on MCMKP, and all of them have proposed heuristic solutions (Akbar et al. 2006, Hifi et al. 2004 and 2006, Khan et al. 2002, Parra-Hernandez and Dimopolous 2005, Moser et al. 1997), except Sbihi (2007). Moser et al. (1997) designed an approach based upon the concept of graceful degradation from the most valuable items based on Lagrange multipliers. It has been observed that Moser et al. (1997) cannot always find a feasible solution when there is one. Khan et al. (2002) tailored the algorithm introduced by Toyoda (1975) for solving the MCMKP. Hifi et al. (2004, 2006) presented two different approximate approaches. The first approach is a guided local-search heuristic in which the trajectories of the solutions were oriented by including a penalty term in the cost function; it penalizes bad aspects of previously visited solutions. The second approach is a reactive local search. It starts with an initial solution, which is improved by an iterative process. The improvement process includes deblocking and degrading procedures in order to escape from local optima and to introduce diversification into the search. Parra-Hernandez and Dimopolous (2005) presents a heuristic that is based on the one given in Pirkul (1987). The authors first reduced MCMKP to a MKP. They solved the linear relaxation of the resulting MKGP, and calculated performance values (called pseudo utility values and resource value coefficients) for each variable. These values were used to find a feasible solution to MCMKP and to improve it. We refer the reader to Kellerer et al. (2004) for a detailed review of heuristic solutions to MCMKP.

The only exact algorithm for MCMKP (Sbihi 2007) finds an optimal solution using B&B. At each B&B node (Sbihi 2007) obtains an upper bound to MCMKP by solving MCKP, which is formed by aggregating knapsack constraints. The computational evaluation presented in (Sbihi 2007) showed that this B&B method was able to solve instances of small and medium sizes with up to 1000 variables, divided into 50 classes (choice constraints) with 20 variables each and up to 7 knapsack constraints. On the other hand, memory requirements prohibited the B&B method from solving larger instances. Furthermore, execution time increased with the number of knapsack constraints. When the number of variables in each class is decreased and the number of knapsack constraints is increased, run time increases for the same number of classes.

2.3. Comparison of bounds

Linear programming (LP), Lagrangian (Gavish and Pirkul 1985, Magazine and Oguz, 1984, Volgenant and Zoon 1990), surrogate (Glover 1968, Osorio et al. 2002), and composite relaxations (Greenberg and Pierskalla 1970) are often used to find lower (upper) bounds for minimization (maximization) problems. The LP relaxation of an IP eliminates the integrality requirements. Lagrangian relaxation (LR) relaxes a set of constraints into the objective function, surrogate relaxation (SR) replaces original constraints $\left(\text{i.e., } Ax \geq b\right)$ with a non-negative linear combination of them $\left(\text{i.e., } sAx \geq sb \text{ for } s^T \in R_+^m\right)$, and composite relaxation (CR) combines both Lagrangian and surrogate relaxations.

Greenberg and Pierskalla (1970) gave the first theoretical analysis of the bounds provided by SR. The most important result of Greenberg and Pierskalla (1970) is that SR provides tighter bounds than LR. Geoffrion (1974) showed that the LR bound is always at least as tight as the LP bound. In addition, Glover (1975) developed surrogate duality theory, which gives strong optimality conditions under which SR has no duality gap; and Karwan and Rardin (1979) investigated the relationship between LR and SR.

Gavish and Pirkul (1985) identified the theoretical relations between LR, SR, and CR for MDKP and proposed new algorithms for obtaining surrogate bounds. Crama and Mazolla (1994) further examined the strength of the bounds obtained through these relaxations and showed two important results. The first result is that CR gives only modest improvement over SR and the second is that, although the bounds derived from LR, SR, or CR are stronger than the bounds obtained from linear relaxation, the improvement in the bound cannot exceed the magnitude of the largest coefficient in the objective function, nor can it exceed one-half of the optimal objective-function value of the linear relaxation. It is important to note that SR provides its most promising results when the number of constraints is very small (Fréville and Hanafi 2005). A recent paper (Ralphs and Galati 2006) illustrated the relationship between LR, Dantzig-Wolfe decomposition (DWD), and cutting plane approaches and presented a framework to improve bounds by integrating dynamic cut generation with LR and DWD, which is well known to be dual to LR (Frangioni 2005).

Lagrangian decomposition (LD) (Guignard and Kim 1987a) relaxes an IP by creating an identical copy of each variable and dualizing the requirement that copies have identical values. LD bounds dominate LR bounds (Guignard and Kim 1987b). However, there is no direct comparison between LD and SR. To our knowledge, there is only one computational study of LD (Guignard et al. 1989); it investigated LD in application to a bi-dimensional KP. Since the number of the Lagrangian multipliers is equal to the number of variables, LD leads to excessive run times and has not previously been shown to be successful in application.

2.4. Valid inequalities

We denote by $K^{\le}$ ($K^{\ge}$) the knapsack constraint in the form of a less (greater)-

than-or-equal-to inequality. The first subsection reviews the literature on the $KPG^{\le}$ polytope and the second summarizes the known valid inequalities for the $KPG^{\ge}$ polytope.

2.4.1. A related polytope

$KP^{\le}$ has been investigated extensively (Balas 1975, Balas and Zemel 1978, Gu et al. 1999, Nemhauser and Wolsey 1988, Weistmantel 1997, Zemel 1989). Balas and Zemel (1978) gave bounds on the lifted coefficients associated with a minimal cover inequalities. Balas (1975), Balas and Zemel (1978), Hammer et al. (1975), Wolsey (1975) and Zemel (1978) proposed a simultaneous lifting procedure to obtain facets. Padberg (1980) introduced (1, *k*)-configurations (i.e., inequalities) for KP.

Lifted cover inequalities, derived from the 0-1 $KP^{\le}$, have been used successfully for solving 0-1 integer problems by cut-and-branch algorithms (Crowder et al. 1983, Gu et al. 1998, Johnson et al. 1985, Gabrel and Minoux 2002). In particular, Crowder et al. (1983) showed that using inequalities for $KP^{\le}$ as cuts for 0-1 integer problems with multiple constraints yields significant computational improvements over pure B&B algorithms.

Several studies (Johnson and Padberg 1981, Nemhauser and Vance 1994, Wolsey 1990) have identified valid inequalities (facets) of the $KPG^{\le}$ polytope. By strengthening valid inequalities for $KP^{\le}$, Wolsey (1990) defined *GUB cover* inequalities for $KPG^{\le}$, and presented specialized implementations of GUB cover inequalities for solving machine-sequencing, generalized-assignment and variable-upper-bounded-flow problems with GUB constraints. Nemhauser and Vance (1994) extended the results of Balas (1975) and Balas and Zemel (1978) and presented a method based on independent sets to lift cover inequalities, obtaining facet-defining inequalities for $\mathrm{KPG}^{\le}$. Glover et al. (1997) devised surrogate-knapsack cuts using a cut-generation method that creates a non-negative linear combination of a knapsack constraint ($\mathrm{K}^{\le}$) with selected bounding inequalities of form $x_j \le 1 \quad j \in J$. A recent study (Zeng and. Richard 2006) analyzed a more general case of $\mathrm{KPG}^{\le}$ in which the right-hand-side of each GUB-like constraint is greater-than-or-equal-to 1. The authors described a lifting procedure for related, generalized cover inequalities using novel, multidimensional super-additive lifting functions that approximate the underlying, exact lifting function from below. Also, a few studies have proposed coefficient reduction methods to tighten the linear relaxation of $\mathrm{KP}^{\le}$ (Johnson et al. 1985, Lougee-Heimer 2001). Our study is different from these in that we study the polyhedral properties of $\mathrm{KPG}^{\ge}$ polytope.

2.4.2. KGP polytope

To our knowledge only Sherali and Lee (1995) have devised a family of valid inequalities (facets) specifically for $\mathrm{KPG}^{\ge}$ polytope. Sherali and Lee (1995) also developed sequential and simultaneous lifting procedures. We refer to reader to Chapter VI for detailed summary of the inequalities devised in Sherali and Lee (1995).

A recent paper (Glover and Sherali 2008) introduced a class of *second-order-cover-cuts* (SOC) for the polytope described by $\mathrm{KP}^{\ge}$ with one additional constraint that defines an upper bound on the sum of all variables. Then, Sherali and Glover (2008) extended the work on SOCs by proposing a new class of *higher-order cover-cuts* (HOC) for $\mathrm{KP}^{\geq}$ with a two-sided bounding constraint on the sum of all variables and a set of two-sided bounding inequalities, each over a unique subset of variables. Let $J = \bigcup_{g \in G} J_g$. For each non-empty subset of indices $J' \subseteq J$, an HOC is given by

$$\sum_{j \in J'} x_j \geq p \tag{2.1}$$

where $p = \min\left\{\sum_{j \in J'} x_j : x \in X\right\}$. Authors presented relationships that identify which of two HOCs dominates the other over the unit hypercube (i.e., $\{x : 0 \leq x \leq 1\}$). Using properties of non-dominated HOCs, Sherali and Glover (2008) focused on generating all non-dominated HOCs by implicitly enumerating all possible $J' \subseteq J$.

2.5. Conclusion

Prior work focused on using LR, SR, or CR to provide bounds in B&B. To our knowledge, no prior research has used B&P-D to provide bounds for MKGP. Moreover, only a few studies have compared bounds provided by LR, SR, or CR computationally and those that have been published are problem specific.

Our study focuses on the polyhedral properties of $\mathrm{KPG}^{\geq}$ polytope. Our research differs from Sherali and Glover (2008) in that we generate valid inequalities to cut off a fractional solution to the linear relaxation of $\mathrm{KPG}^{\geq}$. For this purpose, we establish dominance relationship between inequalities of form (2.1) over the $\mathrm{KPG}^{\geq}$ polytope, present a polynomial-time procedure to generate a non-dominated inequality, describe the conditions under which non-dominated inequalities are facet-defining, and discuss a procedure that lifts sequentially with respect to GUBs, but simultaneously computes lifted coefficients for all variables associated with each GUB.

CHAPTER III

PROBLEM FORMULATION*

This chapter formulates an integer model of the design problem and provides a detailed description of the parameters in the model, fulfilling our first research objective. Section 3.1 formulates the SSDP as a MKGP. Section 3.2 describes the data that reflects the size and scope of an actual application and deals with the practical considerations that are important to ports and waterways in general.

3.1. Sensor system design model formulation

Our model relates four important entities: environmental conditions, sensor combinations, potential sensor locations, and surveillance points. We define an index set E of environmental conditions under which surveillance must be provided; each $e \in E$ denotes a unique (time of day, weather condition) combination, where, for example, the former could be day or night; and the latter, clear, heavy rain or fog. It is possible to install a combination of several types of sensors at the same location; for example, one tv camera, two tv cameras, or a tv camera and an infrared camera could be installed on the same tower. For this reason, we assume that an index set K of sensor combinations can be defined *a priori* as an input to the model. Each "combination" $k \in K$ involves either one sensor type or several. To facilitate presentation, we suppress the generic term

---

"type" in association with a sensor or a sensor combination if ambiguity does not result. We define index set L of potential sensor locations; each $l \in L$ represents a plot of land that can be procured as a site at which a tower could be constructed so that sensors can be installed at appropriate heights. We use c_{kl} to denote the present worth cost of purchasing, installing, and maintaining sensor combination k at location l. We discretize the area to be observed, defining an index set S of surveillance points, each $s \in S$ of which must be observed to assure security (see Section 3.2.4). Although we define each element of notation when we first use it, we summarize frequently used symbols in Table 1 for reader convenience.

Table 1. Notation.

Index sets:	
E	: environmental conditions, which are indexed by $e \in E$
K	: sensor combinations, which are indexed by $k \in K$
L	: potential sensor locations, which are indexed by $l \in L$
S	: surveillance points, which are indexed by $s \in S$
Φ_{kl}^{+}	: subset of (e,s) constraints in (3.5) that have positive coefficients for x_{kl}
Φ_{kl}^{0}	: subset of (e,s) constraints in (3.5) that have zero coefficients for x_{kl}
Parameters:	
c_{kl}	: present worth cost of purchasing, installing, and maintaining sensor combination k at location l
p_{ekls}	: probability that the system, using sensor combination k at location l, will detect an intrusion if one occurs at surveillance point s under condition e
$\overline{p}_{ekls}$	: probability that the system, using sensor combination k at location l, will fail to detect an intrusion if one occurs at surveillance point s under condition e; $\overline{p}_{ekls} = 1 - p_{ekls}$
t_{es}	: maximum acceptable probability for the system to fail to detect an intrusion at surveillance point s under condition e
Decision variables:	
x_{kl}	= 1 if sensor combination k is installed at location l; 0 otherwise
v_{ekls}	: clone of x_{kl} corresponding to constraint (e,s) in (3.5)

We model surveillance capability using p_{ekls}, the probability that the system, using sensor combination k at sensor location l, will detect an intrusion if one occurs at surveillance point s under environmental condition e. The probability that the system, using k at l, will fail to detect an intrusion if one occurs at s under e is given by $\overline{p}_{ekls} = 1 - p_{ekls}$. Section 3.2.5 details how $\overline{p}_{ekls}$ can be calculated.

The probability that the system would not detect an intrusion at surveillance point s under environmental condition e, π_{es}, is the product of the probabilities that the system using all k at all l to observe s under e would fail to detect an intrusion: $\pi_{es} = \prod_{k \in K} \prod_{l \in L} (\overline{p}_{ekls})^{x_{kl}}$ where decision variable $x_{kl} = 1$ if k is located at l, 0 otherwise.

In order to provide sufficient surveillance of s under e, π_{es} should be less than t_{es}, the maximum acceptable probability for the system to *fail to detect an intrusion* at s under e; that is,

$$\pi_{es} = \prod_{k \in K} \prod_{l \in L} (\overline{p}_{ekls})^{x_{kl}} \leq t_{es}.$$

M1, the surveillance system design problem, can now be formulated:

$$Z^*_{M1} = \text{Min} \sum_{k \in K} \sum_{l \in L} c_{kl} x_{kl} \tag{3.1}$$

$$\text{s.t.} \quad \prod_{k \in K} \prod_{l \in L} (\overline{p}_{ekls})^{x_{kl}} \leq t_{es} \qquad e \in E,\ s \in S \tag{3.2}$$

$$\sum_{k \in K} x_{kl} \leq 1 \qquad l \in L \tag{3.3}$$

$$x_{kl} \in \{0,1\} \qquad k \in K,\ l \in L. \tag{3.4}$$

The objective (3.1) is to minimize the total present worth cost of purchasing, installing, and maintaining all sensors in the system. Inequalities (3.2) assure that the required level of surveillance is provided to each surveillance point s under each environmental condition e. Constraints (3.3) allow at most one sensor combination to be installed at each location l. Finally, (3.4) requires all decision variables to be binary.

M1 is a non-linear program, which we now recast in a linear form by transforming constraints (3.2) using logarithms. First, we take the logarithm of each side of constraint (3.2), obtaining

$$\log\left(\prod_{k\in K}\prod_{l\in L}(\overline{p}_{ekls})^{x_{kl}}\right)\leq \log(t_{es}).$$

Continuing,

$$\log\left(\prod_{k\in K}\prod_{l\in L}(\overline{p}_{ekls})^{x_{kl}}\right)=\sum_{k\in K}\sum_{l\in L}\log(\overline{p}_{ekls})^{x_{kl}},$$

so that (3.2) can be expressed as

$$\sum_{k\in K}\sum_{l\in L}\log(\overline{p}_{ekls})^{x_{kl}}\leq \log(t_{es}) \qquad e\in E,\ s\in S.$$

Since $0<\overline{p}_{ekls}\leq 1$ and $0<t_{es}\leq 1$, $\log(t_{es})\leq 0$ and $\log(\overline{p}_{ekls})\leq 0$. Letting $a_{ekls}=-\log(\overline{p}_{ekls})\geq 0$ and $b_{es}=-\log(t_{es})\geq 0$, constraint (3.2) can be re-expressed as:

$$\sum_{k\in K}\sum_{l\in L}a_{ekls}x_{kl}\geq b_{es} \qquad e\in E,\ s\in S. \tag{3.5}$$

So, model (3.1)-(3.4) becomes a linear MKGP:

$$Z^*_{MKGP}=\min\left\{\sum_{k\in K}\sum_{l\in L} c_{kl}x_{kl} : (3.3), (3.4), \text{and } (3.5)\right\}.$$

From this point on, we use (e,s) to denote the knapsack constraint in (3.5) associated with environmental condition e and surveillance point s.

3.2. Test instances

This dissertation focuses on model formulation and solution approach and does not seek to describe a methodology to estimate parameter values in an actual application. Due to security restrictions, we do not have access to actual data that describes any particular port or waterway. However, we generate data that reflects the size and scope of an actual application and deal with the practical considerations that are important to ports and waterways in general. This section describes the test instances that we use to evaluate our solution methods in Chapters IV-VII.

We use the HSC as a test bed. HSC is important because it exemplifies ports and waterways in general and, as the sixth largest port in the world, it handles the largest foreign tonnage among all U.S. ports. (Port of Houston 2008) USCG officers stationed at the Sector Houston-Galveston (SHG) of the Port of Houston provide security through surveillance and patrol boats, which can be dispatched to interdict intruders. The SHG has historically employed television cameras and radar to maintain surveillance, primarily to manage the flow of large commercial vessels.

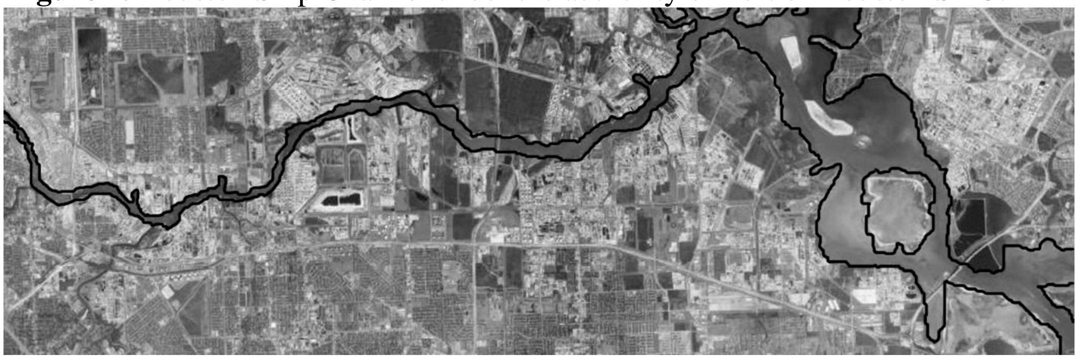
HSC is vulnerable to a number of security threats, which we describe here only briefly and in general terms. The channel's shoreline is home to a huge petro-chemical complex that includes some of the world's largest plants. These critical facilities process and store both hazardous and flammable materials near populous areas. An intruder might try to gain access to the ship channel in a number of ways, perhaps using a small, fast boat.

A vessel enters HSC near Galveston Island and travels in a northwest direction through Galveston Bay. This region is under the authority of an SHG sub-unit based on Galveston Island, the Galveston Marine Safety Unit (GMSU). Passing through Morgans Point, the vessel continues into a region that is under the authority of the SHG, entering a narrower waterway that is oriented in a northwestern direction, then making a 270° turn at Lynchburg Ferry Crossing, and proceeding into an even narrower waterway that meanders in a westward direction to the Houston Turning Basin. The subsection from Morgans Point to Lynchburg Ferry Crossing has no restrictions on boating, so that small pleasure craft, fishing boats, tugs, barges can use it along with large commercial vessels. Boating is restricted on the subsection from Lynchburg Ferry Crossing to the Houston Turning Basin, so that small craft do not have permission to use it.

The ship channel is actually a dredged channel that is (roughly) in the center of the waterways described. Large vessels must travel in this dredged channel, although smaller boats are able to navigate the width of the waterway. Nevertheless, the entire waterway is commonly called HSC without distinguishing the dredged portion.

We do not consider the part of the channel that is under the authority of GMSU because it involves a large body of water (the Galveston Bay) for which radar is the primary means of surveillance. Rather, we focus on the portion of the HSC that is under the authority of Port of Houston SHG. It is 20.84 miles long and its width varies from 0.08 miles to 2.51 miles as depicted by Figure 1. A vulnerable petro-chemical complex lines the shore and residential areas are nearby, heightening the need for surveillance. We generated Figure 1 and other aerial views using satellite images available from Google Earth (Google Earth 2008) and ArcGIS (Esri 2008) software.

Figure 1. Houston Ship Channel under the authority of Port of Houston SHG.



The following subsections describe factors and how we generate parameters a_{ekls}, b_{es}, and c_{kl}.

3.2.1. Environmental conditions

We consider three environmental conditions (time of day, weather condition) to represent the broad range of challenges under which surveillance must be assured: (day, clear), (night, clear), and (day, heavy rain). Since each sensor type provides its mid-range capability for (day, heavy rain), we choose it as Level 1, so with $|E| = 1$, E comprises only (day, heavy rain). A sensor type that provides superior capabilities under one condition may not be useful at all under others. No single environmental condition presents a worst case for all sensor types, so the design model must consider all conditions explicitly. Level 2 of $|E|$ is 3, where E comprises (day, clear), (night, clear), and (day, heavy rain).

3.2.2. Sensor combinations

We consider three types of sensors, each of which has two different ranges: thermal cameras (T) (Thermal Camera 2008) with ranges of 4000m (T1) and 3000m

(T2); image intensification (i.e., night vision) cameras (N) (Night Vision Camera 2008) with ranges of 3000m (N1) and 2500m (N2); and closed circuit television cameras (CCTV) (Closed Circuit Television Camera 2008) with ranges of 2500m (CCTV1) and 2000m (CCTV2).

We define 14 sensor combinations (Table 2); each provides a full 360° field of view and can be installed on a single tower. Combinations that involve two senor types provide complementary capabilities.

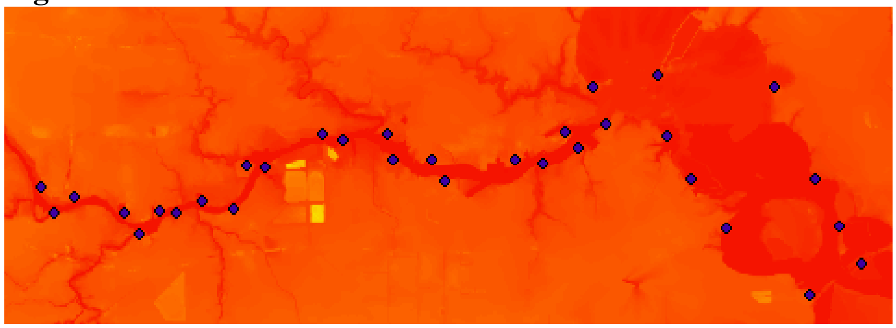
Table 2. Sensor combinations.

k	**Sensor types**
1	T1
2	T2
3	N1
4	N2
5	CCTV1
6	CCTV2
7	T1, CCTV1
8	T1, CCTV2
9	T2, CCTV1
10	T2, CCTV2
11	N1, CCTV1
12	N1, CCTV2
13	N2, CCTV1
14	N2, CCTV2

3.2.3. Potential sensor locations

To identify potential sensor locations, we studied an aerial view of the ship channel. We identified 25 unused plots of land (Figure 2) and added the 7 locations currently used as CCTV locations by SHG, assuming that all 32 plots could be used as sensor locations, either through lease or purchase.

Figure 2. Potential sensor locations.



3.2.4. Discretization

To discretize the channel area, we have drawn a "line of surveillance" across HSC (i.e., perpendicular to the mid-line of the channel) at each 0.5 mile interval and locate a surveillance point at the center of each line. We assume that any sensor that is capable of observing the point would also be able to observe the entire line and its vicinity. This results in $|S| = 42$ for Level 1 of Factor 3. For Level 2 of $|S|$ (i.e., $|S| = 84$), we define two surveillance points on each line, one near the shore at each end of each line, and require that each be observed by sensor(s) located on the same side of the channel. This enhances the capability of detecting intrusion from the shore and provides surveillance from both sides of the channel to assure that a small boat cannot evade detection by hiding behind a large vessel.

3.2.5. Calculating a_{ekls} values

If k cannot provide any surveillance capability under e; if the line of sight from l to s is blocked, for example, by a man-made structure or a terrain feature; or if the straight line distance from l to s, d_{ls}, is beyond the range of k, $\overline{p}_{ekls} = 1$ and $a_{ekls} = 0$.

We use ArcGIS to determine the subset of surveillance points $s \in S_l \subseteq S$ that can be seen in a unobstructed line of sight from each potential sensor location $l \in L$. The relative height of a tower installed at l is important in determining the points that can be observed from that location. We consider three alternative heights: 20m, 40m, and 60m. Figures 3, 4, and 5 show the points (lighter shading) that can be observed in a direct line of sight from towers of 20m, 40m, or 60m height at the point shown at the northeast corner of the channel. Towers of 20m height cannot provide sufficient surveillance capability and towers of 60m are more costly and do not provide substantially better capabilities than towers of 40m height due to elevations and terrain features . Therefore, we use towers of 40m height.

Figure 3. Points that can be observed using a tower of 20m height.

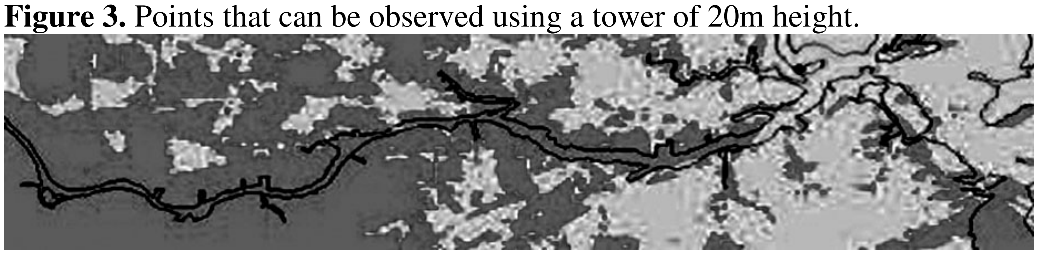


Figure 4. Points that can be observed using a tower of 40m height.

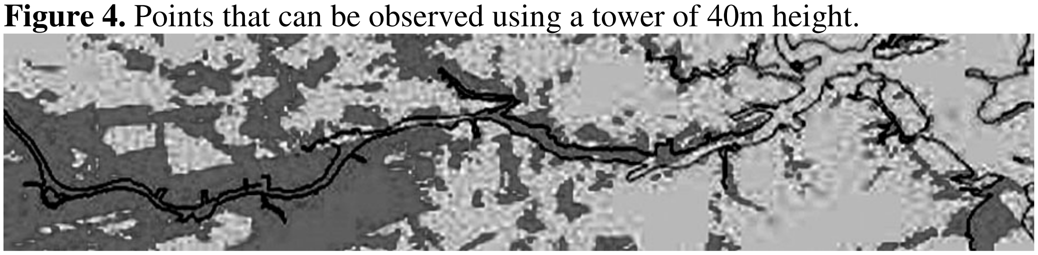
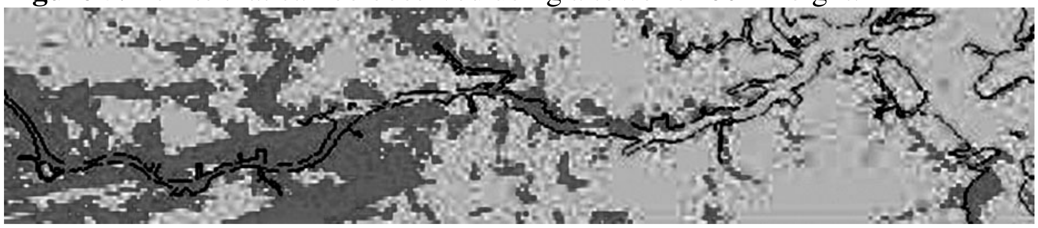


Figure 5. Points that can be observed using a tower of 60m height.



Sensor combination k cannot detect an intrusion at surveillance point s under environmental condition e if no constituent sensor has an unobstructed line of sight, has the necessary range, has sufficient capability under e, or is able to send a positive signal. Let us introduce the following notation:

i : index for sensor type in combination k; $i \in K_k$

d_{ls} : straight line distance from sensor combination mounted on a 40m tower at location l to surveillance point s.

If sensor type i does not have any electro-mechanical problem, it is *operational*; otherwise it is *not operational.* Let W_{ei} be a random variable that has the value 1 if i is operational under environmental condition e; 0 otherwise. The probability that sensor type i is *operational* under e is $\Pr[W_{ei} = 1]$. $\Pr[W_{ei} = 0] = 1 - \Pr[W_{ei} = 1]$ is the probability that sensor type i is *not operational.* To say that i detects an intrusion at surveillance point s correctly means that, given that it is *operational*, it sends a positive signal whenever an intrusion occurs at s and the system interprets the signal properly, perhaps including recognition by a human who monitors a display of sensor signals. Let D_{eils} denote the random variable that has value 1 if the system using sensor type i at location l detects intrusion at s correctly under e, given that i is operational; 0 otherwise. The detection probability is given by $\Pr[D_{eils}=1]$.

The probability $\overline{p}_{ekls}$ can be determined in the following way. Let K_k denote the set of sensor types i in combination k. Combination k cannot detect an intrusion if each sensor $i \in K_k$ is either *not operational* or does not detect an intrusion correctly, given that it is *operational*. We assume that D_{eils} and W_{ei} are mutually independent and that sensor types in K_k work mutually independently. Considering the possibility that subset $\hat{K}_{ekls} \subseteq K_k$ at l is planned to observe s under e but is *not operational*, it can be shown that $\overline{p}_{ekls}$ is given by

$$\overline{p}_{ekls} = \sum_{\hat{K}_{ekls} \subseteq K_k} \left[\prod_{i \in \hat{K}_{ekls}} \left(1-\Pr[W_{ei}=1]\right) \prod_{i \in K_k \setminus \hat{K}_{ekls}} \left(\left(1-\Pr[D_{eils}=1]\right) \Pr[W_{ei}=1] \right) \right].$$

The system detection probability using *operational* sensor i depends on environmental condition, sensor capabilities, and distance from a sensor location to a surveillance point. Detection probability decreases as the distance increases, and can be calculated using $(1-\tilde{e}^{-\kappa_i (d_{ls})^{-\eta_i}})$ (Cavalier et al. 2007); here κ_i and η_i are parameters that represent the decrease in detection probability of sensor type i as d_{ls} increases and $\tilde{e}$ is the Euler's number (we use $\tilde{e}$ since we use e to denote an environmental condition). Since detection probability, $\Pr[D_{eils}=1]$, also depends on e and i, we define parameter ϑ_{ei} for sensor type i, which relates the rate of decrease in $(1-\tilde{e}^{-\kappa_i (d_{ls})^{-\eta_i}})$ to environmental condition e. We use the expression

$(1-\vartheta_{ei})(1-\tilde{e}^{-\kappa_i (d_{ls})^{-\eta_i}})$ to calculate $\Pr[D_{eils}=1]$.

Detection probability decreases rapidly as d_{ls} increases for sensor types N (night vision) and CCTV (closed circuit television) and slowly for type T (thermal). Considering this, we determine (κ_i, η_i) for each i as in Table 3. Also, it is important to note that a sensor cannot provide surveillance for points that are very close to it, as it magnifies such a target too highly to allow effective observation.

Table 3. Parameter values selected for calculating $\bar{p}_{ekls}$.

i	κ_i	η_i	ϑ_{ei}			$\Pr[W_{ei}=0]$		
			Day	**Night**	**Rain**	**Day**	**Night**	**Rain**
T1	3.500	1.000	0.100	0.000	0.050	0.030	0.030	0.045
T2	2.600	1.000	0.100	0.000	0.050	0.030	0.030	0.045
N1	2.000	0.800	1.000	0.000	0.200	0.010	0.010	0.015
N2	1.700	0.750	1.000	0.000	0.200	0.010	0.010	0.015
CCTV1	1.600	0.750	0.000	1.000	0.100	0.010	0.010	0.015
CCTV2	1.350	0.700	0.000	1.000	0.100	0.010	0.010	0.015

Thermal contrasts are enhanced by temperature differences that typically occur several hours after sunset, so T performs best at night (Thermal Imaging 2008). Also, the thermal contrast between a target and its background is enhanced during rain, so T performs better in rain than during a clear day. Since using N during daytime or in very brightly situations is damaging, we assume that N is not used during a clear day (Thermal Imaging 2008). Also, N performs better at night than on a rainy day (Night Vision Camera 2008). A CCTV can be used only during the day but its capability decreases in rain. Based on these considerations we specify the ϑ_{ei} values shown in

Table 3.

We choose failure probabilities, $\Pr[W_{ei} = 0]$, based on sensor characteristics as shown in Table 3. Since high humidity is known to damage sensors, failure probabilities increase during rain.

3.2.6. Calculating b_{es} values

We calculate t_{es} based on the threat level under e and the characteristics of the critical facilities around s. Since it is more difficult to detect an intruder at night or on a rainy day than on a clear day, we assume that an intrusion is more likely to occur under such conditions. On the other hand, the goal of an attack may be to inflict damage not only on a critical facility, but also to nearby residential areas. Since more people are at home, a night attack may inflict more damage. Hence, we assume that an intrusion is more likely to occur at night. We employ this rationale to specify t_{es} values. Based on the characteristics of the critical facilities around *s*, we first calculate t_{es} for night time (i.e., $e = 2$), the environmental condition under which the threat level may be the highest. Then, we calculate t_{es} for $e=1$ and 3.

We consider all critical facilities that are located on the shore of the HSC, including both refineries and chemical plants. We first classify the refineries and chemical plants that are located on the shore of the HSC in nine categories, based on three criteria: flammable/toxic material storage capacities, flammable/toxic material storage capacities of critical facilities close to them, and closeness to other critical facilities. Applying the Analytic Hierarchy Process (Saaty 1980), we determine a normalized weight, w_j, to represent the importance of each category $j \in \{1...9\}$: 0.215, 0.190, 0.158, 0.119, 0.103, 0.077, 0.057, 0.048 and 0.034. Then, we define the crucial distance, r, between each category of facilities and a surveillance point relative to each of five different types of vessels (i.e., threats). Distance is crucial because it is related to the time required to interdict a threat to a facility once detected at s. We assume that the maximum speed that a small boat can travel in the HSC is 35 knots (~40 mph) and that the interdicting force needs to interdict a threat when it is no closer than 4 minutes from the targeted facility. Therefore, a small-boat (i.e., treat) must be interdicted when it is at least $4 \times (40/60) \approx 2.7$ miles away from the facility, and we assign a higher detection probability, t_{es}, if s is further than 2.7 miles from the facility. Detection probability t_{es} can be reduced if the distance from the facility to s exceeds 3.2 miles because it may be more difficult to associate a threat specifically with the facility or, if the distance is less than 2.7 miles, because security forces would not have time to interdict the threat, even if it is detected. For a large vessel, which travels much slower than a small boat, the same amount of time could be provided for interdiction by making the crucial distance from s to the facility smaller.

We assume that the surveillance system should detect an intrusion at each surveillance point with probability at least 0.95 (on average 0.965). Since providing a detection probability higher than 0.98 may be costly, we also assume that the maximum detection probability requirement is 0.98. We specify the detection probability requirement, $1-t_{2s}$, for surveillance points that is within distance r of facilities in categories 1 and 9 (the most important and the least important categories) as shown in Table 4.

Table 4. Detection probability requirements for critical facilities in categories 1 and 9.

Distance	Vessel speed (mph)	Category 1	Category 9
$3.2 \geq r > 2.7$	40	0.980	0.960
$2.7 \geq r > 2.4$	35	0.975	0.957
$2.4 \geq r > 2.0$	30	0.970	0.955
$2.0 \geq r > 1.0$	15	0.965	0.953
$1.0 \geq r > 0.0$	10	0.960	0.950

Let ρ_r^j denote the detection probability requirement for a surveillance point s that is within a distance r of some critical facilities in a category $j \in \{1...9\}$. We assume that ρ_r^j increases linearly with normalized weight of category j and calculate values for $j = 2...8$ using the following equation:

$$\rho_r^j = \rho_r^9 + \left(\rho_r^1 - \rho_r^9\right)\left[\frac{w_j - w_9}{w_1 - w_9}\right]. \tag{3.6}$$

Equation (3.6) scales the difference between the maximum and minimum required detection probabilities $\left(\rho_r^1 - \rho_r^9\right)$ according to the position of w_j on the range $[w_1, w_9]$.

In order to calculate t_{es} for $e = 1, 3$, we first determine normalized weights ϖ_e $e = 1, 2, 3$ associated with the probability of an intrusion under each environmental condition using the Analytic Hierarchy Process: 0.271, 0.339 and 0.305, respectively. The normalized weight is higher if an intrusion is more likely under the associated environmental condition. Therefore, we assume that the detection probability requirement of s under e, $(1-t_{es})$, is linear with normalized weight and calculate it by scaling $(1-t_{2s})$ according to the ratio (ϖ_e/ϖ_2):

$$t_{es} = 1-(1-t_{2s})\left(\frac{\varpi_e}{\varpi_2}\right) \qquad e \in E/\{2\}.$$

3.2.7. Determining c_{kl} values

Parameter c_{kl} gives the cost of purchasing, installing, and maintaining sensor combination k at location l: $c_{kl} = \mathrm{P}_k + I_k + M_k + L_{kl}$, where

P_k : present worth cost of purchasing sensor combination k

I_k : present worth cost of installing sensor combination k

M_k: present worth cost of maintaining sensor combination k over its lifetime (i.e., 5 years)

L_{kl} : present worth cost of the land needed to install sensor combination k at location l.

We obtain the cost of purchasing and installing each sensor combination from equipment manufacturers and adopt the standard practice of using 10% of this cost as an estimate of the annual maintenance cost. We estimate land costs based on average asking prices for plots of similar size in the vicinity of each l.

CHAPTER IV

BRANCH-AND-PRICE DECOMPOSITION

For ease of presentation, in this chapter we consider MKGP with the following structure:

$$Z^*_{MKGP} = \min \sum_{j\in J} c_j x_j \tag{4.1}$$

$$\text{s.t.} \quad \sum_{j\in J} a_{ij} x_j \ge b_i \qquad i \in I \tag{4.2}$$

$$\sum_{j\in J_g} x_j \le 1 \qquad g \in G \tag{4.3}$$

$$x \in \{0,1\}^n \tag{4.4}$$

in which $\bigcup_{g\in G} J_g = J$; $J_g \cap J_h = \varnothing \quad g,h \in G$ and $g \ne h$; $m = |I|$ and $n = |J|$. Row i in (4.2) is a knapsack constraint and row g in (4.3) is a GUB. We use $A = [a_{ij}]_{m\times n}$ to denote the $m \times n$ matrix of coefficients in (4.2) and a^i to denote the vector of coefficients in row i of A. We require $a_{ij}, b_i, c_j \ge 0$ for $i \in I$ and $j \in J$.

The goal of this chapter is to synthesize an effective solution approach. To that end, we explore several B&P-Ds both analytically and computationally. We use the term B&P-D, because it is reflective of Lagrangian Decomposition (Guignard and Kim 1987a, 1987b). As part of our theoretical analysis, we compare the bounds available from B&P-Ds with three alternative relaxations (LR, LD, and SR), and study whether incorporating a surrogate constraint can improve bounds or not. Our second objective is to evaluate a suite of alternatives with the goal of identifying an effective means of implementing B&P-D for solving MKGP. Our third objective is to compare bounds from different decompositions and implementation alternatives computationally to assess the trade-off between the tightness of resulting bounds and the run times required to obtain them.

This chapter comprises five sections. Section 4.1 formulates alternative decompositions. Section 4.2 presents our theoretical analysis of bounds. Section 4.3 proposes several alternative techniques to implement decompositions. Finally, Section 4.4 discusses computational results.

4.1. B&P-D formulations

In this section, we introduce an alternative formulation of MKGP and compare different ways of decomposing MKGP into a master problem (MP) and subproblems (SPs). We begin by creating m clones of each x_j $j \in J$, one for each constraint (4.2). Using y_i to denote the clone of parent x associated with constraint i of (4.2), (4.1)-(4.4) may be re-expressed as CMKG:

$$Z^*_{CMKG} = \min \sum\nolimits_{j \in J} c_j x_j \tag{4.5}$$

s.t. (4.2), (4.3), and (4.4)

$$x - y_i = 0 \qquad i \in I \tag{4.6}$$

$$a^i y_i \geq b_i \qquad i \in I \tag{4.7}$$

$$\sum\nolimits_{j \in J_g} y_{ij} \leq 1 \qquad i \in I,\ g \in G \tag{4.8}$$

$$y_i \in \{0,1\}^n \qquad i \in I \tag{4.9}$$

Using equalities (4.6), we can eliminate x, giving formulation YMKG, which involves only y_i clones:

$$Z^*_{YMKG} = \min \sum_{i \in I} \hat{c}_i y_i \tag{4.10}$$

s.t. (4.7), (4.8), and (4.9)

$$y_{i'} - y_i = 0 \qquad i \in I \setminus \{1\},\ i' = i - 1, \tag{4.11}$$

where $\hat{c}_i$ must be defined such that $\sum_{i \in I} \hat{c}_i = c$ and (4.8) requires y_i for each $i \in I$ to satisfy each GUB constraint, $g \in G$. Note that $Z^*_{MKGP} = Z^*_{CMKG} = Z^*_{YMKG}$.

Dealing with the linear relaxation of (4.7)-(4.11), we decompose YMKG into a MP and SPs in three different ways. Each of the following three subsections presents one of these B&P-D formulations and studies relationships among the bounds these formulations provide. In each case, each SP relates to a specific i (i.e., $SP(i)$) and each MP forms a convex combination of the extreme points of the polytope associated with $SP(i)$. Note that the subscript on B&P-D, MP, RMP, SP, Z^*, $z^*(i)$, $SP(i)$ and $P(i)$ denotes the type of B&P-D formulation.

4.1.1. B&P-D$_1$

Although cloning expands the size of the problem appreciably, the block-diagonal structure of YMKG can be exploited to form a B&P-D by relegating (4.8) and (4.11) to MP$_1$:

$$Z^*_1 = \min \sum_{i \in I} \sum_{p \in P_1(i)} \left(\hat{c}_i y_i^p\right) \lambda_i^p \tag{4.12}$$

$$\text{s.t.} \quad \sum_{p \in P_1(i')} y_{i'}^p \lambda_{i'}^p - \sum_{p \in P_1(i)} y_i^p \lambda_i^p = 0 \qquad i \in I \setminus \{1\},\ i' = i - 1 \tag{4.13}$$

$$\sum_{p\in P_1(i)}\sum_{j\in J_g} y_{ij}^p \lambda_i^p \le 1 \qquad i\in I,\ g\in G \tag{4.14}$$

$$\sum_{p\in P_1(i)} \lambda_i^p = 1 \qquad i\in I \tag{4.15}$$

$$\lambda_i^p \ge 0 \qquad i\in I,\ p\in P_1(i), \tag{4.16}$$

where Z_1^* is the optimal objective function value for MP_1; $P_1(i)$ is the (index) set of extreme points of the polytope associated with SP i, denoted by $SP_1(i)$; λ_i^p is the decision variable associated with the p^{th} extreme point $p\in P_1(i)$; and $y_i^p \in \{0,1\}^n$ denotes the p^{th} extreme point.

We define *m* SPs, in which $SP_1(i)$ compromises (4.9) and knapsack constraint *i* of (4.7). B&P-D_1 treats the knapsacks in (4.7) as being independent but ultimately requires (using (4.11)) all clones of vector x to have the same value. Subproblem i, $SP_1(i)$, is

$$z_1^*(i) = \min\left\{\sum_{g\in G}\sum_{j\in J_g}\left[\hat{c}_{ij} + \left(\alpha_{i'j} - \alpha_{ij}\right) - \beta_{ig}\right] y_{ij} : a^i y_i \ge b_i,\ y_i \in \{0,1\}^n\right\} - \gamma_i,$$

where $i' = i-1$ for $i\in I\setminus\{1\}$,

$z_1^*(i)$: optimal objective function value for $SP_1(i)$,

$\alpha_i \in R^n$: vector of dual variables associated with i^{th} subset of *n* constraints (4.13),

$\beta_i \in R_-^{|G|}$: vector of dual variables associated with i^{th} subset of $|G|$ constraints (4.14),

$\gamma_i \in R$: dual variable associated with the i^{th} convexity constraint (4.15).

Since MP_1 involves a huge number of columns, B&P-D_1 solves a restricted MP_1 (RMP_1), which comprises only a subset of columns in MP_1. Given an optimal solution to RMP$_1$, associated dual variables are incorporated in the objective function coefficients of each $SP_1(i)$, which is then solved to determine if a column can improve the current RMP$_1$ solution. The solution to $SP_1(i)$ generates an improving column if $z_1^*(i) < 0$ and the current solution to MP$_1$ is optimal if $z_1^*(i) \geq 0$ for all $i \in I$.

Dual variables induce values for clones y_{ij} $j \in J$ for which $a_{ij} = 0$: for such variables, the solution to $SP_1(i)$ includes

$$y_{ij} = 1 \text{ if } \left(\hat{c}_{ij} + \left(\alpha_{i'j} - \alpha_{ij}\right) - \beta_{ig}\right) < 0 \text{ and } y_{ij} = 0 \text{ otherwise.} \quad (4.17)$$

At each iteration, we include all improving columns identified by solving SPs in RMP$_1$, which is then re-optimized. We repeat this procedure until no more improving column can be found.

4.1.2. B&P-D$_2$

The second decomposition assigns GUBs (4.8) both to MP$_2$ and to SP$_2$s (e.g., $SP_2(i)$ for $i \in I$). MP$_2$ is the same as MP$_1$, which is given by (4.12)-(4.16), except that instead of $P_1(i)$ it incorporates $P_2(i)$, which is the (index) set of extreme points of the polytope associated with $SP_2(i)$. The optimal solution value of MP$_2$ is Z_2^*. $SP_2(i)$ is given by

$$z_2^*(i) = \min\left\{\sum_{g \in G}\sum_{j \in J_g}\left[\hat{c}_{ij} + \left(\alpha_{i'j} - \alpha_{ij}\right) - \beta_{ig}\right]y_{ij} : a^i y_i \geq b_i; \sum_{j \in J_g} y_{ij} \leq 1, g \in G, y_i \in \{0,1\}^n\right\} - \gamma_i,$$

where $i' = i - 1$ for $i \in I \setminus \{1\}$. $SP_2(i)$ prescribes values for variables of GUB g (i.e., y_{ij} $j \in J_g$) that have $a_{ij} = 0$ for each $j \in J_g$ in knapsack i as follows:

$$y_{i\hat{j}} = 1 \text{ for a } \hat{j} \in \arg\min\left\{j \in \hat{J}_g : \left(\hat{c}_{ij} + \left(\alpha_{i'j} - \alpha_{ij}\right) - \beta_{ig}\right) < 0\right\} \text{ if } \hat{J}_g \neq \emptyset$$

and $y_{ij} = 0$ for each $j \in J_g$ otherwise,

where $\hat{J}_g \subseteq J_g$ such that $\left(\hat{c}_{ij} + \left(\alpha_{i'j} - \alpha_{ij}\right) - \beta_{ig}\right) < 0$ for each $j \in \hat{J}_g$.

The rationale underlying B&P-D$_2$ is that, like $SP_1(i)$, $SP_2(i)$ can be solved in pseudo-polynomial time (Kellerer et al. 2004), and the following proposition shows that including GUBs in SPs provide tighter bounds in comparison with including GUBs in only MP.

Proposition 4.1. B&P-D$_2$ provides tighter bounds than B&P-D$_1$; i.e., $Z_1^* \le Z_2^* \le Z_{MKGP}^*$.

Proof. In order to prove that $Z_1^* \le Z_2^*$, we first show that the optimal solutions of MP$_1$ and MP$_2$ correspond to the intersection of GUB (4.8) polytopes and the convex hulls of SPs.

Let Ω_1 and Ω_2 denote the polytopes associated with feasible regions of MP$_1$ and MP$_2$, respectively. Recall that both MP$_1$ and MP$_2$ are given by (4.12)-(4.16), but differ in the (index) set of extreme points of the SP polytope(s) that they incorporate.

Now, for each $i \in I$, define $\hat{x}_i = \sum_{p \in P_1(i)} y_i^p \lambda_i^p$, in which $\sum_{p \in P_1(i)} \lambda_i^p = 1$ and $\lambda_i^p \ge 0$, so that $\bar{x}_i$ is a convex combination of the extreme points of $SP_1(i)$. After replacing $\sum_{p \in P_1(i)} y_i^p \lambda_i^p$ with $\hat{x}_i$ in MP$_1$,

$$Z_1^* = \min \sum_{i \in I} \hat{c}_i \hat{x}_i$$

$$\text{s.t.} \quad \hat{x}_{i'} - \hat{x}_i = 0 \qquad i \in I \setminus \{1\},\ i' = i - 1 \qquad (4.18)$$

$$\sum_{j \in J_g} \hat{x}_{ij} \le 1 \qquad g \in G,\ i \in I \qquad (4.19)$$

$$\hat{x}_i \in Conv\{p : p \in P_1(i)\} \qquad i \in I \tag{4.20}$$

$$0 \le \hat{x}_i \le 1 \qquad i \in I. \tag{4.21}$$

Constraints (4.18) and (4.20) together imply that each feasible solution of MP_1 is in the intersection of the convex hulls of SPs. Hence, after replacing (4.20) with $\hat{x} \in \bigcap_{i \in I} Conv\{p : p \in P_1(i)\}$, we can drop (4.18) from MP_1 (4.18)-(4.21), and the polytope associated with MP_k for k = 1,2

$$\Omega_k = \Omega_1 \cap \Omega_{k2},$$

where $\Omega_1 = \left\{x \in R_+^n : \sum_{j \in J_g} \hat{x}_j \le 1,\ g \in G;\ 0 \le \hat{x} \le 1\right\}$ and

$\Omega_{k2} = \bigcap_{i \in I} Conv\{p : p \in P_k(i)\}$. Since $\text{Conv}\{p : p \in P_2(i)\} \subseteq \text{Conv}\{p : p \in P_1(i)\}$ for each $i \in I$, $\Omega_2 \subseteq \Omega_1$; so that, $Z_1^* \le Z_2^* \le Z_{MKGP}^*$. ■

4.1.3. B&P-D$_3$

The third decomposition assigns GUBs (4.8) to only SPs. Denoting the (index) set of extreme points of the polytope associated with subproblem $i, SP_3(i)$, by $P_3(i)$, MP_3 is

$$Z_3^* = \min\left\{\sum_{i \in I} \sum_{p \in P_3(i)} (\hat{c}_i y_i^p) \lambda_i^p : (4.13), (4.15) \text{ and } (4.16)\right\}.$$

Letting $i' = i - 1$ for $i \in I \setminus \{1\}$. Subproblem, $SP_3(i)$, is given by

$$z_3^*(i) = \min\left\{\sum_{g \in G} \sum_{j \in J_g} \left[\hat{c}_{ij} + (\alpha_{i'j} - \alpha_{ij})\right] y_{ij} : a^i y_i \ge b_i; \sum_{j \in J_g} y_{ij} \le 1, g \in G,\ y_i \in \{0,1\}^n\right\} - \gamma_i.$$

The rationale underlying B&P-D$_3$ is the definition of the MP polytope associated with each B&P-D. In the proof of the Proposition 4.1, we showed that the feasible region of each MP comprises the points in the intersection of GUB (4.8) polytopes and the convex hulls associated with SPs. However, if we assign GUBs to SPs, the extreme points of SPs, as well as their convex combinations, satisfy GUBs (4.8). Therefore, B&P-D_3 uses GUBs (4.8) to tighten bounds so that including them also in MP does not provide further tightening. By not incorporating any GUBs in MP_3, MP_3 becomes smaller than MP_2, facilitating solution. While B&P-D_2 and B&P-D_3 provide the same bounds, we study B&P-D_2 computationally to determine whether including GUBs in MP_2 leads to dual values that induce the generation of better columns (i.e., SP extreme points) to facilitate solution.

Proposition 4.2. B&P-D_3 provides the same bound as B&P-D_2; i.e.,

$$Z_1^* \le Z_2^* = Z_3^* \le Z_{MKGP}^* .$$

Proof. Since the feasible regions of MP_2 and MP_3, are the same, $\Omega_2 = \Omega_3$, $Z_2^* = Z_3^*$. ■

It is important to note that these three decompositions are not the only alternatives to decompose YMKG. However, an advantage that each of these B&P-Ds offers is that their SPs can be solved in pseudo-polynomial time. Other B&P-Ds may provide tighter bounds than B&P-D_3 but would require each SP to incorporate more than one knapsack and associated GUBs so that solving it would become as hard as solving MKP. With this motivation, we evaluate these three B&P-D formulations computationally in section 4.4.

In each of these B&P-Ds each SP comprises single knapsack constraint; decompositions B&P-D_2 and B&P-D_3 include GUBs, forming $MCKP^{\ge}$ SPs, and B&P-D_1 does not, forming $KP^{\ge}$ SPs. We modify each SP without GUBs to be a less-than-or-equal-to knapsack constraint ($KP^{\le}$) by complementing variables, and then use the

COMBO algorithm (Martello et al. 1999) to solve it. We modify each SP with GUBs to be a $\mathrm{MCKP}^{\leq}$ using a method similar to the one we describe in (Section 5.2.2) and then employ the Mcknap algorithm (Pisinger 1995) to solve it.

4.2. Analysis of bounds

In the first subsection, we compare the strength of the bounds that can be obtained from B&P-D with those of Lagrangian methods: LR and LD. In the second subsection, we study the strength of the lower bound that a surrogate constraint in B&P-D can provide and state the relationships between the bounds that may be obtained from B&P-D with those from SR and CR.

4.2.1. Lagrangian methods

We briefly review relaxation methods in order to establish notation.

Linear Relaxation. The LP of MKGP relaxes integrality restriction $x \in \{0,1\}^n$:

$$Z^*_{LP} = \min\left\{cx : Ax \geq b; \sum\nolimits_{j \in J_g} x_j \leq 1; 0 \leq x \leq 1\right\}.$$

Lagrangian Relaxation. LR(u^r), the LR of MKGP with respect to constraints $Ax \geq b$ using a vector of Lagrange multipliers $u^r \in R^m_+$ is given by

$$Z_{LR}(u^r) = \min\left\{cx + \sum\nolimits_{i \in I} u^r_i (b_i - a^i x) : \sum\nolimits_{j \in J_g} x_j \leq 1 \; g \in G; x \in \{0,1\}^n \right\}.$$

The problem of maximizing $Z_{LR}(u^r)$ over $u^r \in R^m_+$ is called the *Lagrangian Dual*:

$$Z_{LR}(\hat{u}^r) = \max\nolimits_{u^r \in R^m_+} Z_{LR}(u^r),$$

where $\hat{u}^r$ is the vector of *optimal* Lagrange multipliers, which yields the tightest possible bound from LR, $Z_{LR}(\hat{u}^r)$.

Lagrangian Decomposition. Many different LDs of MKGP can be defined. To obtain SPs that are easier to solve than MKGP, as in B&P-D, we consider each knapsack constraint (4.7) as a separate SP. Considering CMKG, for a given vector of Lagrange multipliers $(u_i^d)^T \in R^n$, LD(u^d) is

$$Z_{LD}(u^d) = \min\left\{cx + \sum_{i \in I} u_i^d (y_i - x) : (4.7), (4.8), (4.9); \sum_{j \in J_g} x_j \le 1 \; g \in G; \; x \in \{0,1\}^n\right\}.$$

Using the vector of *optimal* Lagrange multipliers, $\hat{u}^d$, the LD dual, LD$(\hat{u}^d)$, is given by

$$Z_{LD}(\hat{u}^d) = \max_{u^d \in R^{m \times n}} Z_{LD}(u^d).$$

Proposition 4.3 shows that MP$_3$, the master problem of B&P-D$_3$ is the dual of LD$(\hat{u}^d)$, so that $Z_{LD}(\hat{u}^d) = Z_3^*$.

Proposition 4.3. Master problem of B&P-D$_3$ is the dual of LD$(\hat{u}^d)$.

Proof. Let $G_x = \left\{x \in \{0,1\}^n : \sum_{j \in J_g} x_j \le 1 \; g \in G\right\} = \left\{x \in \{0,1\}^n : \hat{\mathbf{G}}x \le 1\right\}$.

$$\begin{aligned}
Z_{LD}(\hat{u}^d) &= \max_{u^d \in R^{m \times n}} Z_{LD}(u^d) \\
&= \max_{u^d \in R^{m \times n}} \left\{\min\left\{cx + \sum_{i \in I} u_i^d (y_i - x) : (4.7), (4.8), (4.9)\right\}\right\} \\
&= \max_{u^d \in R^{m \times n}} \left\{\begin{array}{l} \min\left\{\left(c - \sum_{i \in I} u_i^d\right)x : x \in G_x\right\} \\ + \sum_{i \in I} \min\left\{u_i^d y_i : 4.7(i),\ 4.8(i),\ 4.9(i)\right\} \end{array}\right\} \\
&= \max_{u^d \in R^{m \times n}} \left\{\min\left\{\left(c - \sum_{i \in I} u_i^d\right)x : x \in G_x\right\} + \sum_{i \in I} \min_{p \in P_3(i)} \left\{u_i^d y_i^p\right\}\right\} \\
&= \max_{u^d \in R^{m \times n}} \left\{\begin{array}{l} \min\left\{\left(c - \sum_{i \in I} u_i^d\right)x : x \in G_x\right\} \\ + \sum_{i \in I} \theta_i^1 : \theta_i^1 - u_i^d y_i^p \le 0,\ i = 1 \ldots m,\ p \in P_3(i) \end{array}\right\}.
\end{aligned}$$

Since $\min\left\{\left(c - \sum_{i \in I} u_i\right)x : -\hat{\mathbf{G}}x \ge -1; -x \ge -1; x \ge 0\right\}$

$$= \max\left\{-\theta_i^2 - \theta_i^3 : -\theta_i^2 \hat{\mathbf{G}} - \theta_i^3 + \sum_{i=1}^{m} u_i^d \le c,\ \theta_i^2 \ge 0,\ \theta_i^3 \ge 0\right\}$$

by duality, last equality becomes

$$\max\left\{-\theta_i^2-\theta_i^3+\sum\nolimits_{i\in I}\theta_i^1 : \begin{array}{c}\theta_i^1-u_i^d y_i^p\le 0,\, i=1...m,\ p\in P_3(i)\\ -\theta_i^2\hat{\mathbf{G}}-\theta_i^3+\sum_{i=1}^{m}u_i^d\le c,\,\theta_i^2\ge 0,\,\theta_i^3\ge 0\end{array}\right\}\text{(By duality)}$$

$$=\min\left\{cx:\begin{array}{ll}x-\sum\nolimits_{p\in P_3(i)}y_i^p\lambda_i^p=0 & i\in I;\\ \sum\nolimits_{p\in P_3(i)}\lambda_i^p=1 & i\in I;\\ \lambda_i^P\ge 0\ \ i\in I,\ p\in P_3(i) & \end{array}\right\}.$$

■

Improved bounds facilitate optimizing an IP by allowing more nodes to be fathomed in the B&B search tree. Lagrangian approaches generally use procedures based on subgradient optimization to search for the optimal Lagrange multipliers $\hat{u}^d$. These approaches may not find the optimal multipliers - if they exist - and usually stop with an approximate solution to $Z_{LD}(\hat{u}^d)$. Therefore, Lagrangian methods are not guaranteed to prescribe an optimal solution to LD$(\hat{u}^d)$. However, B&P-D always provides a bound that is as tight as possible since it provides an exact (i.e., optimal) solution to the associated MP. As a result, we have the following corollary.

Corollary 4.4. For $u^d\in R^{m\times n}$, $Z_{LD}(u^d)\le Z_{LD}(\hat{u}^d)=Z_2^*=Z_3^*\le Z_{MKGP}^*$.

Since neither a SP that is $\mathrm{KP}^{\ge}$ nor $\mathrm{MCKP}^{\le}$ has the integrality property (Wilhelm 2001), each B&P-D yields a lower bound that can be tighter than the linear relaxation of MKGP. On the other hand, the tightest bound that LR(u^r) can possibly provide (i.e., using the optimal Lagrange multipliers) is equal to Z_{LP}^* (Geoffrion 1974), since LR(u^r) has the integrality property; i.e.,

$$Conv\left\{x\in R_+^n:\sum\nolimits_{j\in J_g}x_j\le 1\ g\in G; x\in\{0,1\}^n\right\}=\left\{x\in R_+^n:\sum\nolimits_{j\in J_g}x_j\le 1\ g\in G; x\ge 0\right\}.$$

The following proposition relates the bounds provided by the methods described in this subsection.

Proposition 4. 5. $Z_{LR}(u^r) \le Z^*_{LP} = Z_{LR}(\hat{u}^r) \le Z^*_1 \le Z_{LD}(\hat{u}^d) = Z^*_2 = Z^*_3 \le Z^*_{MKGP}$.

Proof. For proof of $Z_{LR}(u^r) \le Z^*_{LP} = Z_{LR}(\hat{u}^r)$ see Geoffrion (1974).

For proof of $Z^*_{LP} \le Z^*_1$ see Guignard and Kim (1987a, 1987b).

$Z^*_1 \le Z_{LD}(\hat{u}^d) = Z^*_2 = Z^*_3 \le Z^*_{MKGP}$ holds by Proposition 4.1 and Corollary 4.4. ■

4.2.2. Surrogate methods

Using multipliers $(u^s)^T \in R^m_+$ and constraints $Ax \ge b$, a surrogate constraint $u^s Ax \ge u^s b$ can be formed. In this subsection, we consider improving the lower bound provided from B&P-D by incorporating a surrogate constraint. We use B&P-D$_3$, because it provides a bound that is tighter than that of B&P-D$_1$ and the same as that of B&P-D$_2$. However, it has fewer constraints in its MP than do B&P-D$_1$ and B&P-D$_2$. We start by briefly reviewing SR and CR in order to establish notation.

Surrogate Relaxation. The SR of MKGP with respect to constraints $Ax \ge b$ is given by

$$Z_{SR}(u^s) = \min \left\{ cx : u^s Ax \ge u^s b; \sum_{j \in J_g} x_j \le 1 \ g \in G; x \in \{0,1\}^n \right\}.$$

The problem of maximizing $Z_{SR}(u^s)$ over all $(u^s)^T \in R^m_+$ is called *Surrogate Dual,* and defined as

$$Z_{SR}(\hat{u}^s) = \max_{(u^s)^T \in R^m_+} Z_{SR}(u^s),$$

where $\hat{u}^s$ is the *optimal* vector of multipliers, which yields the tightest possible SR bound.

Composite Relaxation. The CR of MKGP with respect to constraints $Ax \geq b$ using vectors of Lagrange $u^c \in R_+^m$ and surrogate $(u^s)^T \in R_+^m$ multipliers is given by

$$Z_{CR}(u^c, u^s) = \min \left\{ cx + \sum_{i \in I} u_i^c (b_i - a^i x) : u^s Ax \geq u^s b; \sum_{j \in J_g} x_j \leq 1 \ g \in G; x \in \{0,1\}^n \right\}.$$

The following well known result from Fréville and Hanafi (2005) relates bounds provided by LR, SR and CR.

Proposition 4.6. $Z_{LR}(u^r) \leq Z_{SR}(\hat{u}^s) \leq Z_{CR}(\hat{u}^r, \hat{u}^s) \leq Z^*_{MKGP}$.

To our knowledge, the literature offers no relationship between the bounds provided by LD and either SR or CR. Also, no prior research has investigated combining SR with DWD for LPs or B&P for IPs. For a given vector of surrogate multipliers $(u^s)^T \in R_+^m$, let Z^*_{3S} be the value of the optimal solution to MP_{3S}, which denotes the master problem obtained after incorporating surrogate inequality $u^s Ax \geq u^s b$ in MP_3. The polytope corresponding to the feasible region of MP_{3s}, Ω_{3S} is given by:

$$\Omega_{3S} = \left\{ x \in R_+^n : u^s Ax \geq u^s b;\ x \in \Omega_3 \right\} \qquad \text{(By Proposition 4.1).}$$

The following proposition establishes that incorporating $u^s Ax \geq u^s b$ in MP_3 cannot tighten Z_3^*.

Proposition 4.7. $Z^*_{3S} = Z^*_3$ for any $(u^s)^T \in R_+^m$.

Proof. By definition of the surrogate constraint, for any $(u^s)^T \in R_+^m$,

$$\Omega_3 \subseteq \left\{ x \in R_+^n : Ax \geq b; 0 \leq x \leq 1 \right\} \subseteq \left\{ x \in R_+^n : u^s Ax \geq u^s b;\ 0 \leq x \leq 1 \right\}.$$

For any $(u^s)^T \in R_+^m$, $u^s Ax \geq u^s b$ is redundant with respect to MP_{3S}, because

$$\Omega_{3S} = \left\{ x \in R_+^n : u^s Ax \geq u^s b;\ x \in \Omega_3 \right\} = \Omega_3.$$

Since including the surrogate constraint does not tighten the feasible region, MP_3 and MP_{3S} give the same lower bound for any vector of multipliers $(u^s)^T \in R_+^m$. ■

It can be seen from the proof of Proposition 4.7 that it is not possible to form a surrogate constraint in any B&P-D that is violated by some fractional points that are feasible in MP. Therefore, adding a surrogate constraint to MP cannot tighten the feasible region of any B&P-D.

However, as the following proposition shows, it is possible to improve the B&P-D bound by including the surrogate as a new SP. Now, using $u^s Ax \geq u^s b$, define an additional SP to B&P-D_3, that is $SP_3(u^s)$. The convex hull of the feasible region of $SP_3(u^s)$ is given by

$$\Delta_3(u^s) = Conv\left\{x \in R_+^n : u^s Ax \geq u^s b, \sum_{j \in J_g} x_j \leq 1 \;\; g \in G, x \in \{0,1\}^n\right\}.$$

By augmenting $SP_3(u^s)$ to B&P-D_3, we obtain B&P-$D_3(u^s)$, which has master problem $MP_3(u^s)$, and optimal solution value $Z_3^*(u^s)$.

Proposition 4.8. $Z_3^* \leq Z_3^*(u^s)$.

Proof. The polytope associated with the feasible region of $MP_3(u^s)$ can be written as

$$\Omega_3(u^s) = \Omega_3 \bigcap \Delta_3(u^s). \qquad \text{(By Proposition 4.1)}$$

Since the feasible region of $MP_3(u^s)$ is contained in that of MP_3 (i.e., $\Omega_3(u^s) \subseteq \Omega_3$), $Z_3^* \leq Z_3^*(u^s)$. ■

If $SP_3(u^s)$ has the integrality property, incorporating it as a new SP in B&P-D_3 cannot yield to a tighter feasible region and, therefore, cannot improve the bound of

B&P-D$_3$. Adding an additional SP to B&P-D$_3$ can improve the bound provided by MP$_3$, Z_3^*, if there exists a surrogate multiplier $(u^s)^T \in R_+^m$ such that the convex hull of feasible integer solutions to $SP_3(u^s)$ does not contain any optimal solution of MP$_3$.

CR can provide a tighter bound than B&P-D$_3$ if there exists an optimal multiplier $\hat{u}^s$ such that $Z_3^* < Z_{CR}(\hat{u}^r, \hat{u}^s)$. Using multiplier $\hat{u}^s$ in B&P-D$_3$(u^s), we will get a bound at least as tight as $Z_{CR}(\hat{u}^r, \hat{u}^s)$; that is, $Z_{CR}(\hat{u}^r, \hat{u}^s) \le Z_3^*(\hat{u}^s)$. However, as can also be seen from the following example, such a surrogate multiplier may not exist.

Example 4.1. $Z_{MKGP}^* = \min\ x_1 + 2x_2 + 2x_3 + x_4$

$$\begin{aligned} \text{s.t. } 3x_1 + 3x_2 + 5x_3 + x_4 &\ge 5 \\ x_1 + 4x_2 + 2x_3 + 3x_4 &\ge 4 \\ x_1 + x_2 &\le 1 \\ x_3 + x_4 &\le 1 \\ x_1, x_2, x_3, x_4 &\in \{0,1\} \end{aligned}$$

For $u^s = [u_1^s\ u_2^s]$ a surrogate constraint is given by

$$(3u_1^s + u_2^s)x_1 + (3u_1^s + 4u_2^s)x_2 + (5u_1^s + 2u_2^s)x_3 + (u_1^s + 3u_2^s)x_4 \ge (5u_1^s + 4u_2^s).$$

Case 1: $u_2^s / u_1^s \ge 1/3$. $x_2 = x_4 = 1$ is feasible to SR and $Z_{CR}(\hat{u}^r, u^s) = 3.5$.

Case 2: $u_2^s / u_1^s < 1/3$. $x_1 = x_3 = 1$ is feasible to SR and $Z_{CR}(\hat{u}^r, u^s) = 3$.

So, $Z_{SR}(\hat{u}^s) = 3 \le Z_{CR}(\hat{u}^r, u^s) \le Z_3^* = Z_3^*(\hat{u}^s) = Z_{MKGP}^* = 4$.

On the other hand, if a multiplier u^s exists such that $Z_3^* < Z_3^*(u^s)$, it does not imply that $Z_3^* \le Z_{CR}(\hat{u}^r, u^s)$ since there may still be a feasible point in CR$(\hat{u}^r, u^s)$ whose objective function value is less than Z_3^*. The following corollary summarizes the results related in this section. Therefore, we present it without proof.

Corollary 4.9. For a given optimal vector of multipliers $\hat{u}^r$ and $\hat{s}$,

i. $Z^*_{LP} = Z_{LR}(\hat{u}^r) \le Z_{SR}(\hat{u}^s) \le Z_{CR}(\hat{u}^r, \hat{u}^s) \le Z^*_3(\hat{u}^s)$.

ii. If $\left\{x \in \{0,1\}^n : cx \le Z^*_3; \hat{u}^s Ax \ge \hat{u}^s b; \sum_{j \in J_g} x_j \le 1, g \in G\right\} \ne \varnothing$, then

$$Z^*_{LP} = Z_{LR}(\hat{u}^r) \le Z_{SR}(\hat{u}^s) \le Z_{CR}(\hat{u}^r, \hat{u}^s) \le Z^*_3 \le Z^*_3(\hat{u}^s).$$

4.3. Implementation techniques

This section describes several alternative techniques to implement B&P-Ds. Each of the following three subsections describes one of these three alternatives: cost function, master problem type and surrogate constraint.

4.3.1. Cost assignment alternatives

We evaluate two ways of specifying $\hat{c}_i$ values: the first is the *uniform cost* assignment in which $\hat{c}_i = \left(\frac{1}{m}\right)c$ for $i \in I$, and the second is the *null cost* assignment in which $\hat{c}_1 = c$ and $\hat{c}_i = 0$ for $i \in I \setminus \{1\}$. Although MP has the same optimal solution value in both cases, they result in different objective function coefficients in SPs. Under uniform cost assignment, all related clones have the same cost coefficient; but, under null cost assignment, y_1 is assigned the parent cost c in its objective function and each related clone has a coefficient of 0.

Under *null cost* assignment, equality constraints (4.6) can be replaced by inequality constraints,

$$-y_{i'} + y_i \le 0 \qquad i \in I \setminus \{1\},\ i' = i - 1. \tag{4.6a}$$

No cost is associated with $y_i \quad i \notin I \setminus \{1\}$ and $a^i \geq 0$ for all $i \in I$. Therefore, any component $y_{1j} = 1$ forces $y_{ij} = 1$ for all $i \in I \setminus \{1\}$ through the chain of inequality relationships (4.6a), so that any solution that is optimal satisfies all inequalities (4.6a) at equality. By substituting inequality constraints, we relax RMP and expect that RMP will be made easier to optimize.

4.3.2. RMP formulation

We evaluate different ways of formulating equality constraints $y_{i'} - y_i = 0$ for $i \in I \setminus \{1\}$ and $i' = i - 1$. Letting $i(j)$ denote the index of knapsack (4.2) corresponding to $i(j) \in \arg\max\{a_{ij} : i \in I\}$, constraint $y_{i'} - y_i = 0$ can be re-expressed as

$$y_{i(j)} - y_i = 0 \qquad\qquad i \in I \setminus \{i(j)\}. \tag{4.22}$$

We conjecture that knapsack, which incorporates the largest coefficient a_{ij}, tends to induce x_j to be 1 in (4.1)-(4.4) more than other knapsacks. If we use (4.22) in RMP, the objective function coefficient in $SP(i(j))$ corresponding to $y_{i(j)j}$ includes the dual variable values corresponding to all clones of parent x_j. Hence, by using (4.22) in RMP we aim to provide dual variable values to $SP(i(j))$ that reflect the impact of other SPs in which $a_{ij} > 0$.

Constraint $y_{i'} - y_i = 0$ can also be recast in quite a different form. Variable x_j appears with $a_{ij} > 0$ only in some rows of (4.2). Now, for each variable $x_j \quad j \in J$ define $I_j^+ \subseteq I$ as the index set of constraints (4.2) in which $a_{ij} > 0$. Also, let $I_j^0 = I \setminus I_j^+$. For each $j \in J$, equality constraints $y_{i(j)j} - y_{ij} = 0$ corresponding to $i \in I_j^0$ can be cast in an aggregated form:

$$\left|I_j^0\right| y_{i(j)j} - \sum_{i \in I_j^0} y_{ij} = 0. \tag{4.23}$$

By aggregating constraints we reduce the number of rows in RMP with the goal of reducing the solution time of B&P-D by virtue of dealing with a smaller RMP.

Aggregating constraints does not change the optimal solution value of RMP. Under *null cost* assignment, the cost coefficient of each y_{ij} $i \in I_j^0$ is zero, so that using (4.23) for clones y_{ij} $i \in I_j^0$ cannot increase the value of the optimal solution. Under *uniform cost* assignment, all related clones y_{ij} $i \in I_j^0 \cup I_j^+$ have the same cost coefficient. Clones y_{ij} $i \in I_j^0$ contribute $\left(\left|\hat{I}_j^0\right| \hat{c}_{j(i')} y_{i(j)j}\right)$ to the value of objective function, independent of the values assigned to them. If there exits an optimal solution to RMP that includes (4.22) for all clones of parent x_j, an equal optimal solution value can be obtained using (4.23) in RMP for clones associated with $i \in I_j^0$ instead of (4.22) for these clones associated with $i \in I_j^0$.

Note that none of these alternatives changes either the feasible region or the optimal solution value of the RMP. However, we expect that each will result in a different run time, since each involves a different set of dual variables that are incorporated in SPs.

4.3.3. Surrogate constraint

We evaluate incorporating a surrogate constraint in RMP. As shown in

Proposition 4.7, this does not improve the bound. However, the formulation that includes a surrogate leads to a different set of dual variables and could lead to faster convergence.

4.4. Computational evaluation

This section describes our tests, which we design to address the second and third research objectives: evaluating alternative combinations of decomposition and implementation techniques; and evaluating the trade off between the bounds that the decompositions make available and the run times required to obtain them, respectively.

Each of our test cases is a combination of a decomposition (Section 4.1) and an implemetation technique (Section 4.3). Each level of Factor 1 (F1) designates a decomposition:

1. B&P-D_1 : GUBs in MP + Knapsack SPs ($KP^{\geq}$),
2. B&P-D_2 :GUBs in MP + Multiple-choice knapsack SPs ($MCKP^{\geq}$),
3. B&P-D_3 : no GUBs in MP + $MCKP^{\geq}$.

Each implementation technique is defined as a selection of one level of F2, F3 and F4:

Factor 2 (F2) (cost assignments):

1. *uniform cost* assignment with equality constraints
2. *null cost* assignment with equality constraints
3. *null cost* assignment with inequality constraints

Factor 3 (F3) (RMP formulation):

1. using (4.22) for all clones included in both I_j^+ and I_j^0
2. using (4.6) for all clones included in both I_j^+ and I_j^0

3. using (4.22) for clones included in I_j^+ and (4.23) for these identified by I_j^0

4. using (4.6) for clones included in I_j^+ and (4.23) for these identified by I_j^0

Factor 4 (F4) (surrogate):

1. RMP without any surrogate constraint

2. RMP with surrogate constraint $\sum_{i\in I}\sum_{j\in J}\left(m^{-1}a_{ij}\right)y_{\hat{i}j} \geq \sum_{i\in I} m^{-1}b_i$

We conduct our tests on a Dell PC (OptPlex GX620) with 3.20GZH Dual Core Processor, 2GB RAM, and 160GB hard drive, using CPLEX 11.

The first subsection describes our test instances. The second subsection reports bounds obtained at the root node of each B&P-D and then details computational results.

4.4.1. Test instances

We perform each of our tests on four instances generated as described in Chapter III. The size of each instance depends on four factors: number of environmental conditions $|E|$, sensor combinations $|K|$, potential sensor locations $|L|$, and surveillance points $|S|$. We draw our four instances (Table 5) from that applied problem setting to provide a basis for evaluation. In Table 5, the first column gives the instance number; columns 2-5 give $|E|$, $|K|$, $|L|$, and $|S|$, respectively; and columns 6-8 give the size of each instance in terms of the numbers of binary variables (BVs) and knapsack constraints (KPs), and the number of GUBs (|G|), respectively.

4.4.2. Test results

We begin by describing the content of Tables 6-7. We use the CPLEX B&B algorithm as a benchmark for our B&P-Ds. Table 6 records measures that describe the performance of CPLEX on each of the four test instances. The first column in Table 6 gives the instance number, and the next three report CPLEX results: optimal LP solution value $Z(LP)$; optimal integer solution value $Z(IP)$; and run time (seconds). Each of the last three columns gives a percentage gap relative to the optimal integer solution value: for the bound obtained from the linear relaxation of MKGP $100((Z(IP)-Z(LP))/Z(IP))$, for the optimal root node solution of our B&P-D with $\mathrm{KP}^{\geq}$ $100((Z(IP)-RNS_1)/Z(IP))$, and for the optimal root node solution of our B&P-D with $\mathrm{MCKP}^{\geq}$ $100((Z(IP)-RNS_2)/Z(IP))$.

Table 5. Description of test cases used for evaluating B&PD.

N	|*E*|	|*K*|	|*L*|	|*S*|	BVs	SPs	|*G*|
1	1	7	14	42	98	42	14
2	3	7	14	42	98	126	14
3	3	7	21	42	147	126	21
4	3	14	14	42	196	126	14

Table 6. CPLEX results for the test instances used in evaluating B&P-Ds.

N	*Z*(*LP*)	*Z*(*IP*)	CPLEX time (secs)	$\frac{Z(IP)-Z(LP)}{Z(IP)}$ (%)	$\frac{Z(IP)-RNS_1}{Z(IP)}$ (%)	$\frac{Z(IP)-RNS_2}{Z(IP)}$ (%)
1	2978.41	3805	12.92	21.724%	1.932%	0.000%
2	3628.67	4504	90.61	19.435%	14.549%	0.000%
3	3201.20	4102	8858.55	21.960%	0.098%	0.037%
4	3562.47	4242	523.00	16.019%	0.436%	0.000%

The columns of Table 7 are organized in three groups. The first group gives the level for each factor: F1, F2, and F3. The second and the third groups report the run times at the root node for levels 1 and 2 of F4, respectively. We set a run time limit of 60×10^3 seconds for each test. If the run time limit is reached in solving an instance, we mark the run time columns with “*”.

Table 7. Root node solution times (seconds) for instances 1-4.

Factors			F4 = 1				F4 = 2			
F1	**F2**	**F3**	**N = 1**	**N = 2**	**N = 3**	**N = 4**	**N = 1**	**N = 2**	**N = 3**	**N = 4**
1	1	1	56.002	2013.890	*	*	51.001	2125.640	*	*
1	1	2	28.657	2579.65	*	*	36.986	2585.49	*	*
1	1	3	39.939	1440.01	*	*	30.860	1741.42	*	*
1	1	4	29.078	1984.41	*	*	21.375	1867.70	*	*
1	2	1	126.000	3745.70	*	*	138.172	3584.27	*	*
1	2	2	73.094	4406.91	*	*	108.000	4017.84	*	*
1	2	3	55.047	2404.79	*	*	84.750	2003.99	*	*
1	2	4	68.094	2930.00	*	*	76.469	2792.43	*	*
1	3	1	189.357	4492.29	*	*	193.879	4574.03	*	*
1	3	2	251.428	6900.41	*	*	256.491	7764.87	*	*
1	3	3	152.457	2800.29	*	*	170.661	3576.62	*	*
1	3	4	199.396	5132.01	*	*	181.254	3816.79	*	*
2	1	1	3.079	24.546	2156.97	182.986	2.251	27.858	2269.54	186.011
2	1	2	4.079	34.154	2739.15	253.349	3.735	30.311	2892.80	232.953
2	1	3	2.687	25.586	1099.85	68.626	2.563	30.401	1742.12	82.423
2	1	4	3.125	19.625	2245.38	69.485	3.000	22.937	2028.24	94.079
2	2	1	2.844	47.267	3720.47	476.615	3.063	51.264	3067.69	494.816
2	2	2	5.203	58.103	4037.01	672.609	4.062	72.872	3890.65	641.373
2	2	3	2.578	24.428	2099.94	116.075	2.299	25.046	2042.12	121.88
2	2	4	2.828	36.362	2445.90	182.734	2.563	38.176	2427.52	175.079
2	3	1	4.531	111.963	5218.68	730.838	6.516	121.979	6056.34	724.337
2	3	2	9.954	173.086	8492.96	1206.08	10.344	139.369	9795.92	1432.96
2	3	3	4.047	45.513	2855.45	401.561	6.141	49.982	3436.70	486.479
2	3	4	5.063	79.06	2719.08	528.360	8.376	71.231	3077.46	596.796
3	1	1	2.484	22.062	2972.74	199.955	2.844	19.593	2957.76	178.250
3	1	2	3.422	30.093	2297.35	232.261	3.360	25.358	2479.98	250.799
3	1	3	2.515	13.188	1030.81	64.142	2.297	10.952	1394.15	66.282
3	1	4	2.907	17.390	1390.80	69.673	2.234	17.343	1215.68	76.985
3	2	1	2.140	49.812	4016.83	482.517	3.219	36.609	4442.80	526.127
3	2	2	3.453	50.114	3745.56	703.488	4.250	55.362	3771.81	750.439
3	2	3	2.203	18.878	1701.71	106.721	2.969	19.094	2151.76	98.610
3	2	4	3.125	35.906	2447.55	160.735	3.360	29.601	2434.14	154.173
3	3	1	4.844	95.558	5729.30	718.125	4.125	88.762	6350.54	765.141
3	3	2	8.422	114.527	6422.55	1073.97	8.109	126.605	7049.31	1027.31
3	3	3	4.078	40.124	2825.20	365.266	4.469	39.327	2503.09	405.313
3	3	4	5.204	76.763	3117.97	472.734	8.078	82.106	3231.47	499.281

* exceeds the time limit of 60,000 seconds

4.4.3. Analysis of bounds

All three B&P-Ds provide considerably tighter bounds than the linear relaxation of MKGP (Table 6). Consistent with Proposition 4.1, B&P-Ds with $MCKP^{\geq}$ provide tighter bounds than those with $KP^{\geq}$. For 3 of 4 instances, B&P-Ds with $MCKP^{\geq}$ find the integral solution at the root node. Consistent with Proposition 4.2, including GUBs in MP does not improve the lower bound; and we get the same lower bound for levels 2 and 3 of F1 (B&P-D formulations), which both employ $MCKP^{\geq}$. Furthermore, B&P-Ds with $KP^{\geq}$, require considerably longer run times than both B&P-Ds with $MCKP^{\geq}$ (Table 7). Considering all instances, the minimum time required to find an optimal solution at the root node using the former is at least twice the maximum time required by the latter. Therefore, we do not report results related to solving CMKG using level 1 of F1 (i.e., B&P-D_1) in the following analysis.

Since the cost assignment (F2) does not change the feasible region of RMP, we get the same lower bound for both uniform and null cost assignments. Even though inequality constraint (4.6a) relaxes the feasible region in comparison to equalities (4.6), if an optimal solution exists, it will satisfy constraints (4.6a) at equality. Therefore, lower bounds are the same for each of the three levels of F2 (cost assignment) in combination with the same set of levels of the remaining three factors (implementation techniques).

Each level of F3 (RMP formulation) expresses constraint (4.6) in a different way, but none of them either tightens or relaxes the feasible region of RMP. Therefore, each of the levels of F3 provides the same bound. Consistent with Proposition 4.7, which shows that including a surrogate constraint in RMP does not tighten the feasible region, each of the two levels of F4 (surrogate) provides the same bound.

4.4.4. Analysis of F1 (decomposition formulations)

This section analyzes the effect of F1 with the goal of determining the influence each level has on run time. With this goal we compare the run times of the cases that have the same levels for factors F2, F3 and F4 in application to instances 1-4 (Figures 6-7). Each of the four instances has 16 cases associated with each level of F1. Table 7 shows that using $KP^{\geq}$s (level 1 of F1) requires the longest run time. Using $KP^{\geq}$s leads to larger B&B search trees; at least 7 B&B nodes must be searched to find an optimal solution to instances 1-2, but $MCKP^{\geq}$s (levels 2 and 3 of F1) are able to find an optimal solution at the root node for each instance. Since level 1 of F1 performs so poorly, we do not consider it further.

We compare the three levels of F1 by adding the run times of cases associated with each level. Letting ϖ_i denote the sum of the run times of cases with level *i* of F1 over all test factor combinations, we use $\Delta_{ij} = 100(\varpi_j - \varpi_i)/\varpi_i$ as a criterion to compare the run times of levels *i* and *j*. $\Delta_{ij} > 0$ means that level *i* is $\Delta_{ij}\%$ faster than level *j*.

Over all test cases, levels 2 and 3 require approximately the same run time. To further determine the significance of F1 for run time, we conduct an analysis of variance (ANOVA) using Minitab 15. The objective of this analysis is to test the hypotheses H_0 that a factor has no effect on run time at $\alpha = 0.05$. Consistent with our analysis, ANOVA does not reject H_0 = F1 (excluding level 1) as its level of significance is 0.919. Thus, run times of levels 2 and 3 are not statistically significantly different. Thus, in case of

MCKP$^{\geq}$s, including GUBs also in RMP neither reduces runtime nor tightens the feasible region of RMP. Based on this analysis, we incorporate level 3 of F1 (B&P-D_3) in our default B&P-D implementation.

Figure 6. Total run times required to find an optimal integer solution for instances 1-2.

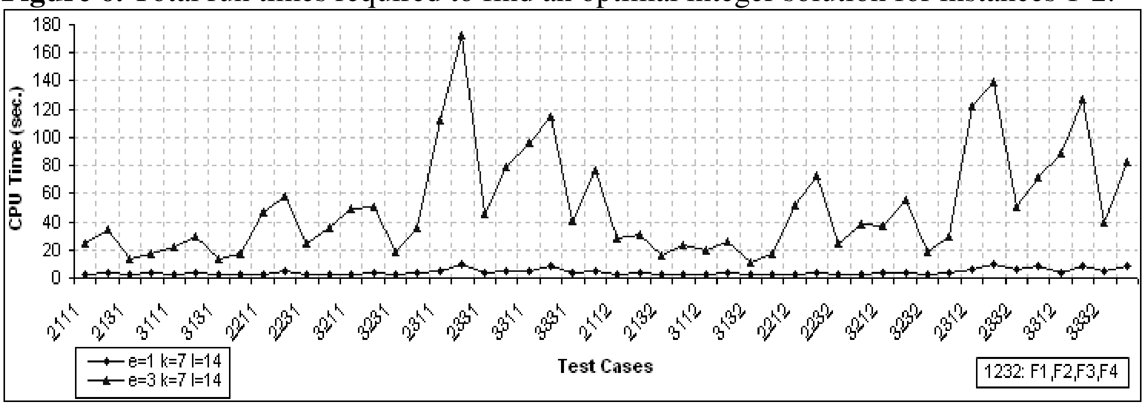
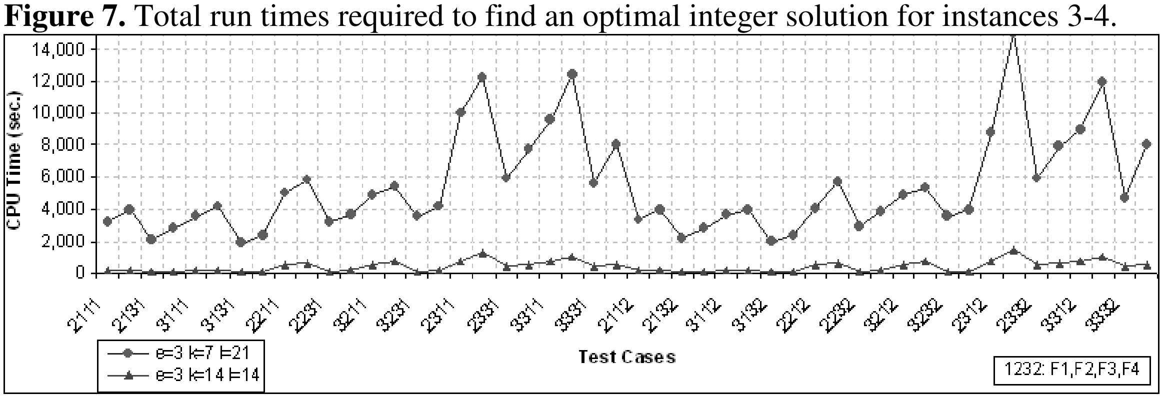


Figure 7. Total run times required to find an optimal integer solution for instances 3-4.



4.4.5. Analysis of F2 (cost assignment)

This section analyzes the effect of F2 with the goal of determining the influence each level has on run time. With this goal, we compare the run times of the cases that have the same set of levels for factors F1, F3 and F4. Each instance has 16 cases associated with each level of F2.

Over all test cases, level 1 requires less run time than level 2 (36% on average), and level 2 requires considerably less run time than level 3 (69% on average) (Figures 6-7). Although re-expressing equality constraints (level 2) as inequalities (level 3) does not change the optimal solution value, it increases the number of feasible solutions to RMP; so that level 3 requires longer run times than do levels 1 and 2. While level 2 assigns a non-zero cost coefficient only to y_1, level 1 assigns the same cost coefficient to each related clone, providing, we expect, more stabilized dual variable values than level 2. We find that level 1 generally requires less run time.

To further determine the significance of F2 on run time, we conduct ANOVA. H_0 = F2 is rejected at the 0.000 *p*-level over all instances, showing that F2 is a significant factor in determining run time. Based on this analysis, we incorporate level 1 (uniform cost assignment) in our default B&P-D implementation.

4.4.6. Analysis of F3 (RMP formulation)

This section analyzes the effect of F3 with the goal of determining the influence each level has on run time. With this goal, we compare the run times of the cases that have the same set of levels for factors F1, F2 and F4. Each of four instances has 12 cases involving each level of F3.

ANOVA rejects H_0 = F3 at *p*-level 0.000 over all instances, showing that this is a statistically significant factor on run time. Over all instances and test cases, level 3 requires less run time than level 4 (25% on average); level 4 requires less run time then level 1 (11% on average); and level 1 requires less run time than level 2 (30% on average) (Figures 6-7). Since levels 3 and 4 incorporate equality constraints only for clones with $a_{ij} > 0$, they result in smaller RMPs, making less challenging to solve. Therefore, levels 3 and 4 are considerably faster than levels 1 and 2. At level 4, two dual values are associated with each clone and the difference between them is used in calculating each cost coefficient in the associated SP. However, such differences are close to each other, especially for the first few RMP iterations, so that columns that are quite different from the ones in the current basis are not generated. Therefore, level 4 requires longer run time than level 3. Based on this analysis, we incorporate level 3 of F3 (using equality (4.21) only for clones with $a_{ij} > 0$) in our default B&P-D implementation.

4.4.7. Analysis of F4 (surrogate constraints)

This section analyzes the effect of F4 with the goal of determining the influence of including a surrogate constraint in RMP on run time. ANOVA does not reject H_0 = F4, since its *p*-level is 0.977, showing that F4 has no statistically significant affect on run time. Moreover, levels 1 and 2 lead to the same number of degenerate iterations and columns entered. The reason for this result is that surrogate constraints are already satisfied in SPs, so their surrogate is redundant in RMP. Based on this analysis, we choose level 1 (no surrogate) in our default B&P-D implementation.

CHAPTER V

BRANCH-AND-PRICE DECOMPOSITION TO DESIGN A SURVEILLANCE SYSTEM FOR PORT AND WATERWAY SECURITY*

The goal of this chapter is an effective solution approach, including a computational evaluation of implementation techniques, for the surveillance system design problem. This chapter fulfills its objective in three sections. Section 5.1 describes our B&P-D approach to design a surveillance system for ports and waterways. Section 5.2 presents alternative implementation techniques to facilitate solution, respectively. Finally, Section 5.3 evaluates alternative B&P-D implementation techniques and describes our computational evaluation.

5.1. B&P-D

B&P-D uses Dantzig-Wolfe Decomposition (DWD) (Wilhelm 2001) to provide lower bounds in a B&B framework. In Chapter IV, we studied various B&P-D formulations that might be applied to MKGP, establishing relationships among the bounds these methods provide. In this section, we describe the B&P-D formulation that requires less run time than others considered in Chapter IV. To our knowledge, such decomposition in conjunction with B&P has not been reported in the literature.

In order to be able to decompose MKGP using DWD, we first transform MKGP into a block diagonal form by generating clones of parent variables x_{kl} $k \in K$, $l \in L$. To exploit the individual knapsack constraints in (3.5), we create $|E|\times|S|$ clones of x_{kl}, one for each (e,s) knapsack. Using v_{ekls} to denote the clone of parent x_{kl} that is associated with (e,s), MKGP may be re-expressed as CMKG:

$$Z^*_{CMKG} = \min \sum_{k\in K} \sum_{l\in L} c_{kl} x_{kl}$$

s.t. (3.3), (3.4), and

$$x_{kl} - v_{ekls} = 0 \qquad e\in E,\ k \in K,\ l \in L,\ s \in S \qquad (5.1)$$

$$\sum_{k\in K} \sum_{l\in L} a_{ekls} v_{ekls} \geq b_{es} \qquad e\in E,\ s \in S \qquad (5.2)$$

$$\sum_{k\in K} v_{ekls} \leq 1 \qquad e\in E,\ l \in L,\ s \in S \qquad (5.3)$$

$$v_{ekls} \in \{0,1\} \qquad e\in E,\ k \in K,\ l \in L,\ s \in S. \qquad (5.4)$$

Remark 5.1. Consider knapsack constraint (e,s) and GUB $l \in L$. If $a_{ekls} = 0$ for all $k \in K$, we say that constraint (e,s) does not contain GUB l; otherwise, constraint (e,s) contains GUB l. If constraint (e,s) does not contain GUB l, fixing any clone v_{ekls} $k \in K$ either to 0 or 1 has no effect on the feasibility of a solution with respect to that (e,s). The values of variables x_{kl} $k \in K$ in a solution are determined by the values of clones v_{ekls} associated with knapsack constraints that contain GUB l. To manage the total number of clones created, we clone variable x_{kl} $k \in K$ only with respect to the knapsack constraints that contain GUB l, creating appropriate v_{ekls} clones. ■

Although cloning expands the size of MKGP significantly, the linear relaxation of CMKG has a block diagonal structure, so that it is amenable to DWD. In order to reduce the size of CMKG with the goal of reducing solution run time, we start by aggregating some of the equality constraints in (5.1) and eliminating x_{kl} variables.

Each x_{kl} variable appears with a non-zero coefficient, $a_{ekls} > 0$, in a subset of (e,s) constraints (3.5). Now, define $\Phi_{kl}^{+}\left(\Phi_{kl}^{0}\right)$ as the index subset of (e,s) constraints (3.5) that have $a_{ekls} > 0$ $\left(a_{ekls} = 0\right)$ associated with variable x_{kl}. Constraints (5.1) can be reformulated as

$$x_{kl} - v_{ekls} = 0 \qquad k \in K,\ l \in L,\ (e,s) \in \Phi_{kl}^{+} \qquad (5.5)$$

$$\sum\nolimits_{(e,s)\in\Phi_{kl}^{0}} v_{ekls} - \left|\Phi_{kl}^{0}\right| x_{kl} = 0 \qquad k \in K,\ l \in L. \qquad (5.6)$$

In (5.5) we use equality constraints only for clones v_{ekls} with $a_{ekls} > 0$, and in (5.6) we aggregate equality constraints corresponding to clones v_{ekls} with $a_{ekls} = 0$. Replacing (5.1) with (5.5) and (5.6) does not expand the feasible regions of CMKG or its linear relaxation, since fixing x_{kl} to either 0 or 1 is feasible with respect to $(e,s) \in \Phi_{kl}^{0}$.

We can now eliminate x_{kl} variables using equalities (5.5) and (5.6), giving a formulation that involves only v_{ekls} variables. For each $k \in K$ and $l \in L$, let $(\bar{e}_{kl}, \bar{s}_{kl}) \in \arg\max_{(e,s)\in\Phi_{kl}^{+}} \{a_{ekls}\}$, breaking ties by choosing the constraint with the lexigraphically smallest index. Suppressing subscripts $(\bar{e}_{kl}, \bar{s}_{kl})$ for convenience, we let $\bar{v}_{kl}$ denote the patriarch of x_{kl}, the clone corresponding to knapsack $(\bar{e}_{kl}, \bar{s}_{kl})$. By substituting patriarch $\bar{v}_{kl}$ for parent x_{kl} in (5.5) and (5.6), we obtain:

$$\bar{v}_{kl} - v_{ekls} = 0 \qquad k \in K, l \in L, (e,s) \in \left\{\Phi_{kl}^{+} \setminus (\bar{e}_{kl}, \bar{s}_{kl})\right\} \qquad (5.7)$$

$$\sum_{(e,s)\in\Phi_{kl}^0} v_{ekls} - \left|\Phi_{kl}^0\right|\bar{v}_{kl} = 0 \qquad k \in K,\ l \in L. \tag{5.8}$$

By replacing constraints (5.1) with equivalents (5.7) and (5.8), CMKG can be re-written as $Z_{CMKG}^* = \min\left\{\sum_{e\in E}\sum_{k\in K}\sum_{l\in L}\sum_{s\in S}\tilde{c}_{ekls}v_{ekls} : (5.2), (5.3), (5.4), (5.7), \text{and } (5.8)\right\}$, where $c_{kl} = \sum_{e\in E}\sum_{s\in S}\tilde{c}_{ekls}$.

In Chapter IV we compare three strategies for assigning values to $\tilde{c}_{ekls}$. Tests described in Chapter IV show that the uniform strategy, which assigns an equal portion of c_{kl} to each clone of parent x_{kl} (i.e., $\tilde{c}_{ekls} = \frac{c_{kl}}{|E||S|}$ for $e \in E$, $k \in K$, $l \in L$, and $s \in S$), performs effectively, so we apply it in this section.

B&P-D decomposes CMKG into a MP, which incorporates (5.7)-(5.8), and $|E|\times|S|$ SPs. $SP(e,s)$ comprises a specific knapsack constraint (e,s) of (5.2) together with GUBs of (5.3) associated with (e,s). B&P-D treats the knapsacks in (5.2) as being independent but requires (using (5.7) and (5.8)) all v_{ekls} clones of parent x_{kl} to have the same value.

Let $\mathrm{P}(e,s)$ denote the (index) set of all (binary) extreme points of the polytope associated with $SP(e,s)$ and let $\lambda_{es}^{\rho} \in \{0,1\}$ be the decision variable in MP associated with extreme point $\rho \in \mathrm{P}(e,s)$. For $e \in E$, $s \in S$, and $\rho \in \mathrm{P}(e,s)$, $v_{ekls}^{\rho} = 1$ if sensor combination k is installed at location l, 0 otherwise. By relaxing the binary requirements on λ_{es}^{ρ}, MP can be expressed as

$$Z^* = \min\sum_{e\in E}\sum_{k\in K}\sum_{l\in L}\sum_{s\in S}\sum_{\rho\in\mathrm{P}(e,s)}\left(\tilde{c}_{ekls}v_{ekls}^{\rho}\right)\lambda_{es}^{\rho} \tag{5.9}$$

$$\text{s.t. } \sum_{\rho\in\overline{\mathrm{P}}_{kl}} \overline{v}_{kl}^{\rho}\overline{\lambda}_{kl}^{\rho} - \sum_{\rho\in\mathrm{P}(e,s)} v_{ekls}^{\rho}\lambda_{es}^{\rho} = 0$$

$$k\in K,\ l\in L,\ (e,s)\in\Phi_{kl}^{+}\setminus\{(\overline{e}_{kl},\overline{s}_{kl})\} \quad (5.10)$$

$$-\left|\Phi_{kl}^{0}\right|\sum_{\rho\in\overline{\mathrm{P}}_{kl}} \overline{v}_{kl}^{\rho}\overline{\lambda}_{kl}^{\rho} + \sum_{(e,s)\in\Phi_{kl}^{0}}\sum_{\rho\in\mathrm{P}(e,s)} v_{ekls}^{\rho}\lambda_{es}^{\rho} = 0 \quad k\in K,\ l\in L \quad (5.11)$$

$$\sum_{\rho\in\mathrm{P}(e,s)}\lambda_{es}^{\rho} = 1 \quad e\in E,\ s\in S \quad (5.12)$$

$$\lambda_{es}^{\rho}\ge 0 \quad e\in E,\ s\in S,\ \rho\in\mathrm{P}(e,s) \quad (5.13)$$

in which $\overline{\mathrm{P}}_{kl}$ denotes the index set of extreme points of the polytope associated with $SP(\overline{e}_{kl},\overline{s}_{kl})$, where $\overline{\lambda}_{kl}^{\rho}$ is the decision variable associated with $\rho\in\overline{\mathrm{P}}_{kl}$.

In general, MP comprises a huge number of columns. Therefore, we solve a restricted master problem (RMP), obtained by replacing $\mathrm{P}(e,s)$ for $e\in E$ and $s\in S$ by one of its subsets, $\hat{\mathrm{P}}(e,s)$.

Three kinds of clones may be defined in $SP(e,s)$, based on the types of dual variables used to calculate the reduced cost associated with each. Given (e,s), let Ω_{es}, Θ_{es}, and Ψ_{es} denote the index set (k,l) of clones v_{ekls} whose reduced cost coefficients are calculated using the dual variables noted below:

Ω_{es} : using the dual variables α_{ekls} and β_{kl} corresponding to (5.10) and (5.11), respectively (i.e., $(k,l)\in\Omega_{es}$ if $(\overline{e}_{kl},\overline{s}_{kl})=(e,s)$);

Θ_{es} : using the dual variable α_{ekls} corresponding to (5.10) (i.e., $(k,l)\in\Theta_{es}$ if $(e,s)\in\Phi_{kl}^{+}\setminus\{(\overline{e}_{kl},\overline{s}_{kl})\}$);

Ψ_{es} : using the dual variable β_{kl} corresponding to (5.11) (i.e., $(k,l)\in\Psi_{es}$ if $(e,s)\in\Phi_{kl}^{0}$).

The generic form of $SP(e,s)$ is

$$z^*(e,s) = -\gamma_{es} + \min \left\{ \begin{array}{l} \sum_{(k,l)\in\Omega_{es}} \left(\tilde{c}_{ekls} - \sum\limits_{(e,s)\in\Phi_{kl}^{+}\setminus(\bar{e}_{kl},\bar{s}_{kl})} \alpha_{ekls} + \left|\Phi_{kl}^{0}\right| \beta_{kl} \right) v_{ekls} + \sum_{(k,l)\in\Theta_{es}} \left(\tilde{c}_{ekls} + \alpha_{ekls} \right) v_{ekls} \\ + \sum_{(k,l)\in\Psi_{es}} \left(\tilde{c}_{ekls} - \beta_{kl} \right) v_{ekls} \end{array} \right\}$$

$$\text{s.t.} \quad \sum_{k\in K} \sum_{l\in L} a_{ekls} v_{ekls} \geq b_{es}$$

$$\sum_{k\in K} v_{ekls} \leq 1 \qquad\qquad l \in L$$

$$v_{ekls} \in \{0,1\} \qquad\qquad k \in K,\ l \in L,$$

where γ_{es} is the dual variable corresponding to the associated convexity constraint (5.12).

Let $\hat{c}_{ekls}$ denote the objective function coefficient of v_{ekls} in $SP(e,s)$. For clones v_{ekls} with $a_{ekls} = 0$, $v_{ekls} = 1$ if $\hat{c}_{ekls} < 0$ and if setting $v_{ekls} = 1$ is feasible with respect to GUBs of (5.3) associated with (e,s); otherwise $v_{ekls} = 0$.

We start with a set of columns that form an initial basic feasible solution (Section 5.2.1) and solve RMP using the primal simplex method. Given an optimal solution to RMP, dual variables α_{ekls}, β_{kl}, and γ_{es} are incorporated in the objective function of each SP, which is solved in an attempt to identify a column that can improve the current RMP solution. The solution to $SP(e,s)$ generates an improving column if $z^*(e,s) < 0$. At each iteration, we include all improving columns identified by solving all SPs in RMP, which is then re-optimized. This process is iterated until $z^*(e,s) \geq 0$ for all (e,s), indicating that the current RMP solution is optimal. We manage the column pool in standard ways (Wilhelm 2001).

5.2. Implementation of B&P-D

This section presents alternative techniques to implement B&P-D. The first subsection devises a heuristic to determine an initial basic feasible solution for the associated RMP, the second presents an effective method for solving SPs, the third describes alternative branching rules, and the fourth mentions the criterion we use to select a node for branching.

5.2.1. Initial basic feasible solution

We now devise a GRASP (greedy randomized adaptive search procedure) (Feo and Resende 1995, Chardaire et al. 2001) to find a set of columns that form an initial basic feasible solution for RMP at each B&B node. Our heuristic actually solves MKGP ((3.1), (3.3)-(3.5)) with certain x_{kl} variables fixed to either 0 or 1 by the branching rule at the associated node in the B&B tree. Then, we generate columns for RMP by fixing all clones of x_{kl} to 1(0) if x_{kl} is prescribed the value 1(0) in the heuristic solution. The heuristic has two phases: construction and local search. The construction heuristic (CH) finds a feasible solution to MKGP, and the improvement heuristic (IH) searches for a less costly feasible solution.

A feasible solution can be found in polynomial time for the 0-1 multidimensional knapsack problem or the multi-choice (single) knapsack problem, if one exists. On the other hand, Moser et al. (1997) mentioned that finding a feasible solution to the MMCKP (i.e, MKGP) requires testing combinations of variables; in the worst case, each possible combination must be tested, so that finding a feasible solution is equivalent to solving MMCKP. The authors did not show that finding a feasible solution to MMCKP is NP-hard.

Proposition 5.1 shows that finding a feasible solution to MKGP is NP-hard. Therefore, our CH is not able to guarantee finding a feasible solution, even if one exits. In such a case, we use phase I of the two-phase simplex method (Bazaraa et al. 1990) to find an initial basic feasible solution to RMP. Phase I is a linear program in terms of λ_{es}^{ρ} and artificial variables in RMP. B&P-D generates a set of columns to prescribe an optimal phase I solution, and we use these columns to form an initial basic feasible solution to RMP.

Proposition 5.1. Finding a feasible solution to MKGP is NP-hard.

Proof: Given a solution to MKGP, we can verify whether it is feasible or not in polynomial time (i.e., $O(n(m+|G|)$). Consequently, finding a feasible solution to MKGP is in class NP.

We now reduce the 3-Partition problem (Garey and Johnson 1979) to MKGP. 3-Partition can be described as follows: given a set $Q=\{q_1,...,q_{3t}\}$ of positive integers and a positive integer T such that $T/4 < q_h < T/2$ for all $1 \le h \le 3t$, and $\sum_{h=1}^{3t} q_h = tT$, does there exist a partition of Q into subsets $Q_1, Q_2, ..., Q_t$ such that $\sum_{q_h \in Q_u} q_h = T$ for all $1 \le u \le t$? The solution to 3-Partition is "yes" if and only if MKGP, as given by

$$\min\left\{\begin{aligned} & cx: \sum_{g\in G} q_g x_{gi} \ge T,\, i\in\{1,...,t\}; \\ & \qquad \sum_{g\in G} x_{gi} \ge 3,\, i\in\{(t+1),...,2t\}; \sum_{j\in J_g} x_{gj} \le 1 \;\; g\in G;\; x_{gj}\in\{0,1\} \;\; j\in J \end{aligned}\right\},$$

has a feasible solution. Thus, finding a feasible solution to MKGP is NP-hard. ■

CH (detailed in Figure 8) comprises two steps ([7c]-[16c] and [18c]-[29c], respectively). In step 1, we randomly select variables to fix to 1 without violating any GUB (3.3). If the step 1 solution violates any (e,s) constraint, step 2 attempts to find a

Figure 8. Construction heuristic.

Input: *An MKGP instance and* a parameter, δ
Output: *A feasible solution for MKGP*

//Initialization
$\hat{K}\hat{L} \leftarrow \varnothing$, $\hat{L} \leftarrow L$ and $\hat{E}\hat{S} \leftarrow \{(e,s) : e \in E, s \in S\}$, num_iter $\leftarrow 0$
$C \leftarrow \{x_{kl} : k \in K, l \in \hat{L}\}$ and $C_R \leftarrow \varnothing$
$\hat{b} \leftarrow b$
Calculate utility values u_{kl} for each $x_{kl} \in C$
//Step 1: *Fix a variable in each GUB*
while $C \neq \varnothing$ and $\hat{E}\hat{S} \neq \varnothing$ **do**
    Sort C in non-increasing order of u values
    Select the first $m \leftarrow \max\{\lfloor \delta|C| \rfloor, 1\}$ variables in C; $x_{kl}^1, x_{kl}^2 \ldots x_{kl}^m$
    $C_R \leftarrow \{x_{kl}^1, x_{kl}^2 \ldots x_{kl}^m\}$
    Randomly select $x_{\bar{k}\bar{l}} \in C_R$
        $\hat{K}\hat{L} \leftarrow \hat{K}\hat{L} \cup \{x_{\bar{k}\bar{l}}\}$
        $C \leftarrow C / \{x_{k\bar{l}} : k \in K\}$
        $\hat{L} \leftarrow \hat{L} / \{\bar{l}\}$
    Modify $\hat{b}_{es}$ for all $e \in E, s \in S$ and $\hat{E}\hat{S}$
    Recalculate the utility values u_{kl} for each $x_{kl} \in C$
//Step 2: *Search a feasible solution*
while $\hat{E}\hat{S} \neq \varnothing$ and num_iter < max_iter
    $R \leftarrow \sum_{(e,s) \in \hat{E}\hat{S}} \hat{b}_{es}$
    $(\hat{e},\hat{s}) \leftarrow \arg\max_{(e,s) \in \hat{E}\hat{S}} \{\hat{b}_{es}\}$
    Randomly select $\hat{k}$ and $\hat{l}$ such that $x_{\hat{k}\hat{l}} \in \hat{K}\hat{L}$
    $\hat{K}\hat{L} \leftarrow \hat{K}\hat{L} \setminus \{x_{\hat{k}\hat{l}}\}$
    Modify $\hat{b}_{es}$ for all $e \in E, s \in S$ and $\hat{E}\hat{S}$
    Calculate infeasibility $R_k \leftarrow \sum_{(e,s) \in \hat{E}\hat{S}} \max\{0, \hat{b}_{es} - a_{ek\hat{l}s}\}$ for each $k \in K \setminus \{\hat{k}\}$
    if $\min_{k \in K \setminus \{k'\}} R_k < R$ **then** $\hat{k} \leftarrow \arg\min_{k \in K \setminus \{\hat{k}\}} R$
    else $\hat{k} \leftarrow \arg\max_{k \in K} a_{\hat{e}k\hat{l}\hat{s}}$
    $\hat{K}\hat{L} \leftarrow \hat{K}\hat{L} \cup \{x_{\hat{k}\hat{l}}\}$
    Modify $\hat{b}_{es}$ for all $e \in E, s \in S$ and $\hat{E}\hat{S}$ and num_iter $\leftarrow$ num_iter + 1

feasible solution. Let $\hat{K}\hat{L} = \{x_{kl} : x_{kl} = 1\, k \in K, l \in L\}$ be the set of variables fixed to 1 by CH, $\hat{L} = \{l : x_{kl} \notin \hat{K}\hat{L} \text{ for any } k \in K\}$ be the index set of GUBs in (3.3) that include no variable fixed to 1 by CH, and $\hat{E}\hat{S} = \{(e,s) : \hat{b}_{es} > 0\}$, in which $\hat{b}_{es} = b_{es} - \sum_{x_{kl} \in \hat{K}\hat{L}} a_{ekls} > 0$, be the index set of (e,s) constraints in (3.5) for which $\hat{K}\hat{L}$ does not define a feasible solution. Line [2c] initializes by setting $\hat{K}\hat{L} = \varnothing$, $\hat{L} = L$, and $\hat{E}\hat{S} = \{(e,s) : e \in E, s \in S\}$. Let the set of candidate variables, $C = \{x_{kl} : k \in K, l \in \hat{L}\}$, comprise all free variables that can be fixed to 1 (individually) without violating any GUB (3.3). The selection of the next variable to fix to 1 starts by sorting [8c] variables in C in non-increasing order of their utility values, $u_{kl} = \sum_{(e,s) \in \hat{E}\hat{S}} \min\{1, a_{ekls} / \hat{b}_{es}\}$ and then selecting [9c] the first m variables $(m = \max\{1, \lceil \delta |C| \rceil\})$ to form a restricted set of candidates, C_R [10c], where $\delta \in [0,1]$ is the GRASP parameter that determines the size of C_R. A variable in C_R is selected at random [11c] and fixed to 1. Update $\hat{K}\hat{L}$, C, $\hat{L}$, $\hat{b}_{es}$ for $e \in E$, $s \in S$, and $\hat{E}\hat{S}$; and u_{kl} for $x_{kl} \in C$, respectively ([12c]-[16c]). Step 1 is repeated until either C or $\hat{E}\hat{S}$ is empty.

Step 2 ([18c]-[29c]) measures the total infeasibility associated with $\hat{K}\hat{L}$ using $\sum_{(e,s) \in \hat{E}\hat{S}} \hat{b}_{es}$ [19c] and identifies constraint $(\hat{e}, \hat{s}) \in \arg\max_{(e,s) \in \hat{E}\hat{S}} \{\hat{b}_{es}\}$ as the most violated one, breaking ties arbitrarily. At each iteration of step 2, the most violated constraint $(\hat{e}, \hat{s})$ is first determined [20c]; then, $x_{\hat{k}\hat{l}} \in \hat{K}\hat{L}$ is selected randomly [21c]. Variable $x_{\hat{k}\hat{l}}$, which was fixed to 1 by step 1, is now fixed to 0 [22c]. If setting a variable $x_{k\hat{l}}$ $k \in K \setminus \{\hat{k}\}$ to 1 would reduce the infeasibility of $(\hat{e}, \hat{s})$, it is fixed to 1 ([23c]-[25c]); otherwise, $x_{k\hat{l}} \in \arg\max_{k \in K}\{a_{\hat{e}k\hat{l}\hat{s}}\}$ is fixed to 1 [26c]. Then, $\hat{K}\hat{L}$, $\hat{b}_{es}$ for $e \in E$ and $s \in S$, and $\hat{E}\hat{S}$ are updated ([27c]-[28c]). This process is repeated until a feasible solution is found or the maximum number of iterations (i.e., max_iter) is reached. Then, we use IH (detailed in Figure 9) to improve the feasible solution found.

Figure 9. Improvement heuristic.

1i $\mathrm{H} \leftarrow L \setminus \hat{L}$
2i **while** $\mathrm{H} \neq \varnothing$
3i   Randomly select $l' \in \mathrm{H}$
4i   Select k' such that $x_{k'l'} \in \hat{K}\hat{L}$
5i $\hat{K}\hat{L} \leftarrow \hat{K}\hat{L} \setminus \{x_{k'l'}\}$
6i   Modify $\hat{b}_{es}$ for all $e \in E, s \in S$ and $\hat{E}\hat{S}$
7i   **if** $\hat{E}\hat{S} = \varnothing$ **then** $\hat{L} \leftarrow \hat{L} \cup \{l'\}$, $\mathrm{H} \leftarrow L \setminus \hat{L}$
8i   **else** $\bar{k} \leftarrow k'$
9i   **for** $k \in K \setminus \{k'\}$ **do**
10i   *feasibility* $\leftarrow$ *true*
11i   **for** $\forall\, (e,s) \in \hat{E}\hat{S}$ **do**
12i   **if** $\hat{b}_{es} - a_{ekl's} > 0$ **then** *feasibility* $\leftarrow$ *false*
13i   **if** (*feasibility* = *true* and $c_{kl'} < c_{\bar{k}l'}$) **then** $\bar{k} \leftarrow k$
14i   **if** $\bar{k} \neq k'$ **then**
15i $k' \leftarrow \bar{k}$, $\hat{E}\hat{S} \leftarrow \varnothing$
16i $\mathrm{H} \leftarrow \left(L \setminus \hat{L}\right) \setminus \{l'\}$, $\hat{K}\hat{L} \leftarrow \hat{K}\hat{L} \cup \{x_{k'l'}\}$
17i   **else** $\mathrm{H} \leftarrow \mathrm{H} \setminus \{l'\}$
18i   Modify $\hat{b}_{es}$ for all $e \in E$ and $s \in S$

The set of GUBs (3.3) that include a variable fixed to 1 by CH is $L \setminus \hat{L}$. We initialize IH with $\mathrm{H} = L \setminus \hat{L}$ [1i]. IH considers each $l \in \mathrm{H}$ in random order [3i]. Let l' be the index of the randomly selected GUB (3.3) and $x_{k'l'}$ be the variable that is fixed to

1 by CH [4i]. If fixing $x_{k'l'}$ to 0 does not violate any constraint in (3.5), it is fixed to 0 [5i]-[7i]. Otherwise, the search considers variables $x_{kl'}$ $k \in K \setminus \{k'\}$ ([8i]-[13i]), and if a less costly variable that does not violate any constraint in (3.5) is found [14i], it is fixed 1, $x_{k'l'}$ is fixed to 0 [15i], and H is updated accordingly $\mathrm{H} = (L \setminus \hat{L}) \setminus \{l'\}$ [16i]; otherwise, $\mathrm{H} = \mathrm{H} \setminus \{l'\}$ [17i]. This process is repeated until $\mathrm{H} = \varnothing$.

At B&B node j, $F_j^1 \subseteq L$ denotes the index set of GUBs (5.3) that include a variable fixed to 1, and $F_j^0 \subseteq L$ denotes the index set l of GUBs (5.3) in which all variables are fixed to 0. At each node j of the B&B tree, we generate $|L| - |F_j^1| - |F_j^0|$ initial basic feasible solutions.

5.2.2. Subproblem solution

We cast $SP(e,s)$ as a $\mathrm{MCKP}^{\le}$ for each $e \in E$ and $s \in S$. Although $\mathrm{MCKP}^{\le}$ is NP-hard, it can be solved in pseudo-polynomial time (Kellerer et al. 2004). We use Pisinger`s algorithm (Pisinger 1995) to solve each SP. This algorithm first finds an initial feasible solution to $\mathrm{MCKP}^{\le}$ and then uses a dynamic programming algorithm to solve $\mathrm{MCKP}^{\le}$. This algorithm was devised to solve a problem in the form that requires exactly one item from each GUB to be prescribed, so that profit is maximized while maintaining feasibility with respect to the capacity (i.e., knapsack) constraint (i.e., a less-than-or-equal-to constraint). At each node in the B&B tree, we fix the clones in all SPs that have been fixed to 0 or 1 by the branching rule and put each $SP(e,s)$ in the $\mathrm{MCKP}^{\le}$ form as follows in order to use Pisinger`s algorithm. First, we modify each GUB $l \in L$

whose variables (i.e., v_{ekls} $k \in K$) each has a positive cost coefficient in the objective function of $SP(e,s)$. We reformulate each such GUB to be an equality constraint by adding a dummy variable with a zero coefficient in the constraint (e,s) and in the objective function, so that the dummy variable does not affect constraint satisfaction or the value of $z^*(e,s)$. A dummy variable need not be included in any GUB $l \in L$ that has at least one variable with non-positive cost coefficient in the objective function of $SP(e,s)$. Assigning the value of 1 to a variable with the most negative cost provides a better solution than assigning the value of 1 to a dummy variable, since coefficients of (e,s) are non-negative and each $SP(e,s)$ minimizes cost; thus, it is already satisfied as equality at the optimal solution. Next, we recast the objective to be maximization and knapsack (e,s) to be a less-than-or-equal-to constraint as described in Kellerer et al. (2004). Let $\hat{c}_{ekls}$ be the reduced cost coefficient of variable v_{ekls} in $SP(e,s)$; $\hat{c}_{els}^{\max} = \max\{\max_{k \in K}\{\hat{c}_{ekls}\},0\}+1$ and $a_{els}^{\max} = \max_{k \in K}\{a_{ekls}\}$ for each $l \in L$. Coefficients of the recast $SP(e,s)$ are as follows:

i. Objective function coefficients: $\bar{c}_{ekls} = \hat{c}_{els}^{\max} - \hat{c}_{ekls}$ for $k \in K$ and $l \in L$;
$\hat{c}_{els}^{\max}$ for the dummy variable in GUB $l \in L$, if one exists

ii. Technological coefficients: $\bar{a}_{ekls} = a_{els}^{\max} - a_{ekls}$ for $k \in K$ and $l \in L$;
$a_{els}^{\max}$ for the dummy variable in GUB $l \in L$, if one exists

iii. Right-hand-side coefficients: $\bar{b}_{es} = \sum_{l \in L} a_{els}^{\max} - b_{es}$.

5.2.3. Branching rule

We evaluate three branching rules. Let $\bar{x}$ be the optimal (fractional) solution to

RMP at a B&B node. The first rule (B1) branches on the most fractional variable, $\bar{x}_{k'l'} = \sum_{p \in \bar{P}_{k'l'}} \bar{v}_{k'l'} \bar{\lambda}^p_{k'l'}$ such that $k'l' \in \arg\min_{k \in K, l \in L} \left| \bar{x}_{kl} - 0.5 \right|$. We create two new B&B nodes (i.e., children): the left child with $\bar{x}_{k'l'} = 0$ and the right child with $\bar{x}_{k'l'} = 1$.

The second rule (B2) branches on the variables $\bar{x}_{kl'}$ $k \in K$ in the GUB l' that includes the most fractional variable. We create $|K|+1$ child nodes: the k^{th} child has $x_{kl'} = 1$ and other variables that are in GUB l' equal to 0; the $(|K|+1)^{st}$ child requires all variables that are in GUB l' to be 0; i.e., $\sum_{k \in K} x_{kl'} = 0$. This branching rule has the advantage that fixing the variables associated with a GUB reduces the number of free variables in RMP more than B1 does, with the hope that resulting RMPs will be less challenging to solve.

The third rule (B3) invokes special order set (SOS) branching. Let $\bar{K}_{l'} \subseteq K$ be the index set of free variables in GUB l' that are not fixed to 0 at the current B&B node, and $\bar{K}^f_{l'} \subseteq \bar{K}_{l'}$ be the index set of free variables in GUB l' that have fractional values in the optimal solution of the corresponding RMP. It is important to note that $\bar{K}_{l'} \setminus \bar{K}^f_{l'}$ is the index set of free variables that have values 0 in the optimal solution of RMP and that $|\bar{K}^f_{l'}| = 0$ if the optimal solution of RMP is integral. When $|\bar{K}^f_{l'}| > 0$, B3 involves two cases (recall that the most fractional variable has indices k'):

Case 1. $|\bar{K}^f_{l'}| = 1$. The left child requires $\bar{x}_{k'l'} = 0$; and the right child, $\bar{x}_{k'l'} = 1$.

Case 2. $|\bar{K}^f_{l'}| \geq 2$. Let

$$\tilde{k} = \begin{cases} (k'+k'')/2 & \text{if } |\,\overline{K}_{l'}^{f}\,| = 2 \text{ such that } \overline{K}_{l'}^{f} = \{k', k''\} \\ \text{median of } \overline{K}_{l'}^{f} & \text{if } |\,\overline{K}_{l'}^{f}\,| \geq 3 \end{cases}.$$

Define $\overline{K}_{\leq \tilde{k}} = \{k : k \leq \tilde{k}, k \in \overline{K}_{l'}\}$ and $\overline{K}_{> \tilde{k}} = \{k : k > \tilde{k}, k \in \overline{K}_{l'}\}$.

The left child requires $\sum_{k \in \overline{K}_{\leq \tilde{k}}} x_{kl'} = 0$; and the right child, $\sum_{k \in \overline{K}_{> \tilde{k}}} x_{kl'} = 0$.

B3 is the same as B1 under the condition of case 1. Under the condition of case 2, B3 has the advantage that, by fixing more than one variable, we expect that it will make RMP less challenging to solve than B1. At each level it creates fewer child nodes than B2.

When branching fixes a variable to 1, other variables in the associated GUB are fixed to 0. Also, whenever a variable is fixed (i.e., either to 0 or 1), related clones in all SPs are fixed to the same value.

5.2.4. Node selection

We invoke the best bound criterion to select the next node to explore in the B&B search. Prior studies have demonstrated that this criterion typically finds an optimal solution in less time and explores fewer nodes in the B&B tree than does the depth-first node selection strategy.

5.3. Computational evaluation

This section describes our test results. We conduct our tests on a Dell PC (OptPlex GX620) with 3.20GZH Dual Core Processor, 2GB RAM, and 160GB hard drive, using CPLEX 11.

We design our tests to achieve two goals. The first goal is to define a default set of implementation alternatives to facilitate B&P-D. In order to achieve this goal, we first compare the performances of the three branching rules (B1, B2, and B3) and select the branching rule that requires the least run time as our default branching rule. Then, in order to evaluate the leverage on run time provided by a good initial solution at each B&B node, we compare the run times required if both CH and IH are used (i.e., CIH) with those required if only the CH is used. The second goal is to evaluate the computational efficacy of B&P-D. For this purpose we benchmark our B&P-D with the B&B routine of CPLEX and analyze the influence of parameters (i.e., experimental factors) on run time. We now begin by describing test instances.

5.3.1. Test instances

Using the HSC as a test bed, in Chapter III we generate instances considering sensor characteristics and the practical considerations that are important to ports and waterways. We perform each of our tests on 16 instances (see Table 8) generated as described in Chapter III. We design instances that involve four factors: numbers of environmental conditions $|E|$, sensor combinations $|K|$, potential sensor locations $|L|$, and surveillance points $|S|$. Three of these factors each has two Levels (see Table 8); $|L|$ has four levels. Level 1(2) of $|E|$ is 1(3). Level 1(2) of $|K|$ is 7(14). Level 1(2) of $|S|$ is 42(84). Levels 1-4 of $|L|$ are 14, 21, 26, and 32, respectively. Since the sensor locations in Level 1 of $|L|$ cannot provide the required level of surveillance to all surveillance points that constitute Level 2 of $|S|$, we use Levels 1 and 2 of $|L|$ in combination with Level 1 of $|S|$ and Levels 3 and 4 of $|L|$ in combination with Level 2 of $|S|$.

Table 8. Description of test instances - HSC.

Instance no	Factors				# of binary variables	# of knapsack constraints	# of GUBs
	$\|E\|$	$\|K\|$	$\|L\|$	$\|S\|$			
1	1	7	14	42	98	42	14
2	1	7	21	42	147	42	21
3	1	14	14	42	196	42	14
4	1	14	21	42	294	42	21
5	3	7	14	42	98	126	14
6	3	7	21	42	147	126	21
7	3	14	14	42	196	126	14
8	3	14	21	42	294	126	21
9	1	7	26	84	182	84	26
10	1	7	32	84	224	84	32
11	1	14	26	84	364	84	26
12	1	14	32	84	448	84	32
13	3	7	26	84	182	252	26
14	3	7	32	84	224	252	32
15	3	14	26	84	364	252	26
16	3	14	32	84	448	252	32

5.3.2. Branching rules and heuristics

Figures 10 and 11 report computational results using CH or CIH in combination with branching rules B1, B2, or B3. 50% of the instances are optimized in the root node. If CH is used with branching rules B1, B2, and B3: on average, B3 is 5.54% faster than B1, and B1 is 60.77% faster than B2; on average, B3 requires 0.5 fewer nodes than B1, and B2 searches over more B&B nodes in all instances than either B1 or B3. If CIH is used with branching rules B1, B2, and B3: on average, B3 is 33.60% faster than B1, and B1 is 58.63% faster than B2; on average, B3 requires 0.75 fewer nodes than B1, and B2 searches over more B&B nodes in all instances than either B1 or B3.

In general, difference between the number of nodes required by B1 and B3 is very small, but using B3 can be advantageous in solving large instances. B3 fixes more than one variable upon branching, resulting in a smaller RMP that requires less computational effort. The advantage provided by a smaller RMP may be substantial; for example, on instance 16 (the largest instance) B1 and B3 each require only three B&B nodes, but B3 runs considerably faster than B1.

Figure 10. Comparison of branching rules B1, B2, and B3.

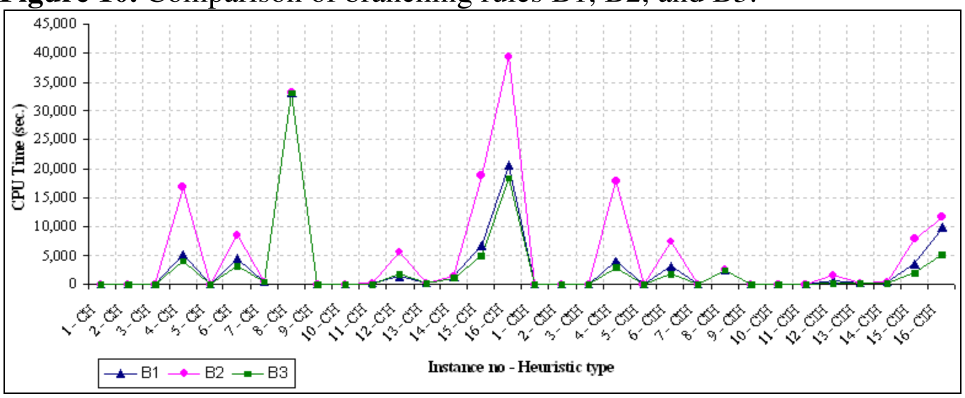


Figure 11. Comparison of CH and CIH.

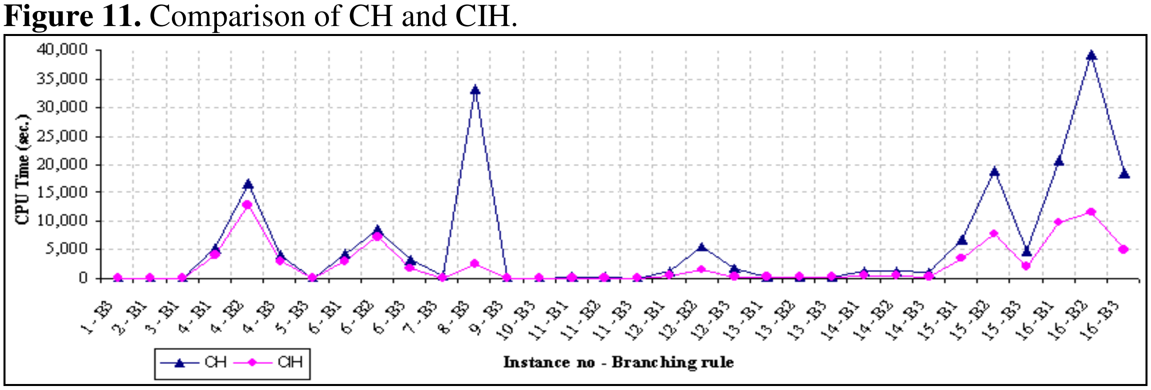


Figure 11 compares CH with CIH. If an optimal solution is found at the root node, it is same for B1, B2, and B3. Hence, in that case we only report the solution associated with B3. This comparison shows that IH has a significant effect, reducing run time in all instances (e.g. on instance 8 CIH requires 2494.44 seconds, and CH requires 33147.5 seconds). Based on these results, we select the CIH in combination with branching rule B3 as the default combination.

5.3.3. Computational evaluation of B&P-D

In this subsection we evaluate the efficacy of B&P-D. For this purpose we benchmark our default B&P-D combination (B&P-D with CIH and branching rule B3) with CPLEX B&B and analyze the effect of experimental factors on run times.

Table 9 reports results from our default combination. The first column in each table gives the instance number (N) (see Table 8) and the next three report results at the root node: the heuristic solution value (HSV), optimal solution value (RNS), and the time required to solve to optimality. The last six columns give results associated with solving CMKG: number of simplex iterations needed for RMP to reach optimality, number of degenerate RMP iterations, total number of generated columns entered, total number of B&B nodes searched, time needed for all RMP simplex iterations, and the total CPU run time to prescribe an optimal integer solution. First we benchmark these results with CPLEX.

5.3.4. Benchmarking

To benchmark our default combination of implementation techniques, we compare it with the B&B routine of CPLEX. Table 10 gives the results. The first column in Table 10 gives the instance number, and the next six report CPLEX results: number of simplex iterations needed to reach optimality (or, number of simplex iterations completed in 60,000 seconds, if CPLEX is terminated because our time limit is

Table 9. B&P with CIH and B3.

N	HSV	RNS	RNS time (sec)	# of simp. iter.	# of deg. iter.	Total # of cols. ent.	# of nodes	RMP sol. time (sec)	CPU (sec)
1	3828	3805.000	2.500	16983	129	1222	1	2.359	2.500
2	3217	2989.000	8.781	31396	325	1949	1	8.485	8.781
3	3663	3500.000	14.172	58326	171	2950	1	13.716	14.172
4	3091	2738.500	1170.660	3129801	1217	32105	5	2887.200	2893.990
5	4504	4504.000	13.188	31788	267	3425	1	12.625	13.188
6	4302	4100.499	1030.810	1413847	222	19808	3	1880.260	1882.540
7	4242	4242.000	64.158	80529	111	6155	1	63.535	64.158
8	4113	3810.00	2494.450	794695	203	12006	1	2492.720	2494.440
9	5936	5881.000	1.906	9653	38	918	1	1.638	1.906
10	5755	5732.000	3.688	15892	74	1411	1	3.344	3.688
11	5472	5368.499	7.562	45066	220	4922	3	12.574	13.968
12	5370	5040.999	153.800	483553	754	15848	5	270.161	278.208
13	7657	7556.499	113.737	313833	629	14449	7	208.522	230.049
14	8061	7778.666	292.142	333212	284	10802	3	370.026	373.533
15	7131	6969.125	1850.190	1006215	632	20966	3	2067.870	2074.620
16	6858	6829.990	4182.720	1462211	697	30202	3	5141.250	5151.070

Table 10. CPLEX results for HSC instances.

N	# of simp. iter.	# of B&B nodes	$Z(LP)$	MBB	$Z(IP)$	CPLEX time (s)	$\frac{Z(IP)-MBB}{Z(IP)}$ (%)	$\frac{Z(IP)-Z(LP)}{Z(IP)}$ (%)	$\frac{Z(IP)-RNS}{Z(IP)}$ (%)
1	191794	106630	2978.41	3805.00	3805	12.9	0.000	21.724	0.000
2	4699267	2135618	2360.44	2989.00	2989	327.9	0.000	21.029	0.000
3	459297	232269	2871.39	3500.00	3500	36.4	0.000	17.960	0.000
4	13859541	59841137	2105.49	2745.00	2745	14039.2	0.000	23.297	0.237
5	1405934	441217	3628.67	4504.00	4504	90.6	0.000	19.435	0.000
6	11471891	35050870	3201.20	4102.00	4102	8858.6	0.000	21.960	0.037
7	6347377	1880246	3562.47	4242.00	4242	523.0	0.000	16.019	0.000
8	65092128	137266419	3045.26	3570.27	3810	60000.0	6.292	20.072	0.000
9	22756694	14294251	4569.67	5881.00	5881	2554.3	0.000	22.298	0.000
10	41869208	239891803	4356.43	5612.69	5732	60000.2	2.081	23.998	0.000
11	43991562	205864226	4190.08	5155.13	5384	60000.3	4.251	22.175	0.288
12	42464790	177343880	3794.84	4561.10	5066	60000.1	9.966	25.092	0.494
13	71659718	133389729	5542.44	7582.00	7582	20571.1	0.000	26.900	0.336
14	40837981	132713241	5415.97	6826.11	7782	60000.0	12.283%	30.404	0.043
15	42699815	135053281	5250.16	6196.39	6970	60000.0	11.099%	24.675	0.013
16	39708547	108500672	4992.27	5845.97	6833	60000.0	14.445%	26.939	0.040

reached), total number of B&B nodes searched, optimal LP solution value ($Z(LP)$), best bound obtained at the termination of CPLEX (MBB), optimal (integral) solution value $Z(IP)$, and run time. The last three columns give the gaps associated with the three lower bounds, showing how far each lower bound is from the optimal solution: the percentage of the gap associated with the best bound obtained at the termination of CPLEX, $100((Z(IP)-MBB)/Z(IP))$; with the bound obtained from the linear relaxation of MKGP, $100((Z(IP)-Z(LP)/Z(IP))$; and with the optimal root node solution of B&P, $100((Z(IP)-RNS)/Z(IP))$. CPLEX exceeds the time limit of 60,000 seconds in 7 of the 16 instances, but our B&P-D prescribes optimal solutions for all 16 instances. B&P-D is faster than CPLEX on the remaining 9 instances by 90% on average. It is important to note that, on all instances, solving RMP accounts for 99.9% of the total CPU time (i.e., SPs require a small portion of run time). MBB values associated with the instances for which CPLEX exceeds 60,000 seconds are smaller (i.e., weaker) than the lower bounds found by our B&P-D at the root node of the B&B tree. Our B&P-D yields a tighter bound than the linear relaxation of MKGP on all instances. Thus, the lower bound obtained at the root node of B&P-D dominates the lower bound obtained from the linear relaxation.

5.3.5. Run time vs. parameters

In this section we evaluate the effect of each parameter (i.e., experimental factor) on run time. Run time to solve CMKG increases as levels of $|E|$, $|K|$, and $|L|$ increase (i.e, Table 9). This is expected, since the number of clones increases with $|E|$, $|K|$, and $|L|$, leading to more challenging RMPs. The number of SPs also increases with $|E|$.

Furthermore, the number of variables in MKGP increases with $|K|$ and $|L|$, requiring more decisions; thus increasing run time.

As seen from Tables 9-10, the most important effect on run time is $|L|$; for example, instances 3 and 4 differ only in their respective values of $|L|$, but their CPU run times are quite different. The reason is that increasing $|L|$ provides more opportunities to locate sensor combinations and increases the number of variables x_{kl} common to different SPs, so the problem becomes more challenging to solve. For example; the difference between the $|L|$ values of instances 3 and 4 is seven, but each additional sensor location increases the number of common variables in approximately 24 constraints. It is important to note that, although the difference between the $|L|$ values of instances 11 and 12 is six, the difference in run time is not as large as that between instances 3 and 4. The reason is that for $|S|$ = 84, we assume that a sensor only observes surveillance points located on the same side of the channel, so each of these additional sensor locations increases the number of common variables in at most 24 constraints.

One might expect that run time always increases with $|S|$, since both the number of clones and the number of SPs increase with $|S|$. If we fix $|L|$ and increase $|S|$, the number of feasible solutions may be decrease, perhaps to the point of rendering the instance infeasible. Therefore, as $|S|$ increases, $|L|$ must be increased in order to satisfy the detection probability required at each $s \in S$. However, Table 9 shows that some instances in which $|S|$ = 84, require less run time than corresponding instances in which $|S|$ = 42 (i.e, instances 4 and 12; instances 6 and 14). The reason is that there is another factor affecting run time: $|L_{eks}|$, the number of potential locations from which k can provide some capability to observe s under e. As $|L_{eks}|$ decreases, the number of variables x_{kl} common to different SPs and the number of clones corresponding to each parent x_{kl} decrease, so the problem becomes less challenging for B&P-D to solve (i.e., $|S| = 84$). Sensors located on either side of the HSC can observe each surveillance point for instances in which $|S| = 42$. However, a sensor that observes a surveillance point must be located on the same side of the channel for instances in which $|S| = 84$, essentially decomposing the problem into two independent components, one associated with each side of the channel. Therefore, for the instances in which $|S| = 84$, fewer surveillance points can be observed from each location l than for the instances in which $|S| = 42$. Thus, for $|S| = 84$, the average value of $|L_{eks}|$ and the number of GUBs associated with each $SP(e,s)$ are both less than for $|S| = 42$.

To further determine the significance of the experimental factors on total run time, we generate 16 more instances defining two new levels for $|S|$ (i.e, $|S| = 22$ and $|S| = 32$) and then conduct ANOVA. The objective of this analysis is to test, at an $\alpha = 0.05$ level, the hypotheses H_0 that a factor or an interaction of factors has no affect on run time versus the alternative H_A that it does. Tests $H_0 = |E|$ and $H_0 = |L|$ are rejected at 0.000 p-level, and $H_0 = |K|$ is rejected at 0.001 p-level. However, for $H_0 = |S|$, the p-level is 0.111, so this hypothesis cannot be rejected. We thus conclude that factors $|E|$, $|K|$, and $|L|$ have significant effects on run time. Furthermore, interactions between these three factors have a significant effect on run time, since $H_0 = |E|\times|L|$, $H_0 = |E|\times|K|$, $H_0 = |L|\times|K|$, and $H_0 = |E|\times|K|\times|L|$ are rejected at p-levels 0.00, 0.005, 0.001, and 0.005, respectively.

CHAPTER VI

KNAPSACK PROBLEM WITH GENERALIZED UPPER BOUND CONSTRAINTS: A POLYHEDRAL STUDY AND COMPUTATION

Chapter V describes a computational evaluation of a B&P-D approach to design a surveillance system, employing the HSC as a test bed. B&P-D is more effective than classical B&B. However, its run time increases with the number of GUBs and the number of variables in each GUB. With the hope of developing a more effective method to solve $\text{MKPG}^{\geq}$ (i.e., MKPG with greater-than-equal-to knapsack constraint), this section defines valid inequalities (facets) for the $\text{KPG}^{\geq}$ polytope.

We consider the $\text{KPG}^{\geq}$ problem, which comprises a knapsack in the form of a greater-than-equal-to constraint and (disjoint) GUBs:

$$Z^*_{KPG^{\geq}} = \min\{cx : x \in X\},$$

where $X = \left\{x \in \{0,1\}^n : \sum_{g \in G} \sum_{j \in J_g} a_j x_j \geq b,\ \sum_{j \in J_g} x_j \leq 1 \quad g \in G\right\}$,

$J = \bigcup_{g \in G} J_g$, and $J_{g'} \cap J_{g''} = \varnothing$ for $g' \neq g'' \in G$.

Each (index) set defined in this section is an (index) subset of either J or G. To facilitate presentation, we use the expression "variable (GUB) in a set" instead of the more lengthy, but more accurate, "index of variable (GUB) in a set" if ambiguity will not result. For $g \in G$, define index $j(g)$ such that $j(g) \in \arg\max\{a_j : j \in J_g\}$. We invoke three assumptions:

Assumption 6.1. $b \geq 0$ and $a_j \geq 0 \quad j \in J$.

Assumption 6.2. $\sum_{g \in G} a_{j(g)} \geq b$.

Assumption 6.3. $\text{conv}(X)$ is a full-dimensional polytope.

Since arbitrarily signed coefficients b and a_j $j \in J$ can be transformed into an equivalent form with $b \geq 0$ and $a_j \geq 0$ $j \in J$ (Johnson and Padberg 1981, Sherali and Lee 1995), Assumption 6.1 imposes no loss of generality. If Assumption 6.2 does not hold, X is infeasible. If $\text{conv}(X)$ is not full-dimensional, it can be modified so that it is (Sherali and Lee 1995); hence, Assumption 6.3 introduces no loss of generality.

This chapter has seven objectives. The first objective is a family of valid inequalities for $\text{conv}(X)$ and the second is a polynomial-time procedure to generate them. The third objective is a set of dominance relationships for these inequalities and the fourth is the necessary and sufficient conditions for a non-dominated inequality to define a facet of $\text{conv}(X)$. The fifth objective is a lifting procedure to tighten valid inequalities that are not facets and the sixth is a separation procedure to generate a valid inequality to cut off a fractional solution to the linear relaxation of $\text{KPG}^{\geq}$. The seventh objective is a computational evaluation of a branch-and-cut approach that uses these inequalities in solving the multidimensional $\text{KPG}^{\geq}$ (i.e., $\text{MKPG}^{\geq}$).

The remainder of the chapter is organized as follows. Section 6.1 reviews known valid inequalities (facets) of $\text{conv}(X)$. Sections 6.2-6.8 address objectives 1-7, respectively. Section 6.2 derives a family of valid inequalities for $\text{conv}(X)$ and Section 6.3 develops a procedure to generate them. Section 6.4 discusses dominance relationships for these inequalities and Section 6.5 establishes necessary and sufficient conditions for a non-dominated inequality to define a facet of $\text{conv}(X)$. Section 6.6 presents a lifting procedure to further tighten the valid inequalities. Section 6.7 devises a separation procedure to generate a valid inequality to separate a fractional optimal solution to a linear relaxation of $\text{KPG}^{\geq}$. Section 6.8 evaluates the efficacy of our cuts in application to solve $\text{MKPG}^{\geq}$.

6.1. The $\text{KPG}^{\geq}$ polytope

To our knowledge, only Sherali and Lee (1995) has devised a family of facets specifically for $\text{conv}(X)$. We now summarize the results of Sherali and Lee (1995), providing a level of detail that is sufficient to allow us to show how our contributions differ.

Proposition 6.1. $\dim(\text{conv}(X)) = |J| - |G_0|$, where $G_0 = \left\{\hat{g} \in G : \sum_{g \in G \setminus \{\hat{g}\}} a_{j(g)} < b\right\}$.

The following two propositions from Sherali and Lee (1995), state the trivial facets of $\text{conv}(X)$.

Proposition 6.2. For each $g \in G$ and $j \in J_g \setminus \{j(g)\}$, $x_j \geq 0$ is a facet of $\text{conv}(X)$.

Proposition 6.3. GUB constraints $\sum_{j \in J_g} x_j \leq 1 \quad g \in G$ are facets of $\text{conv}(X)$.

Sherali and Lee (1995) defined a generalization of the well-known minimal cover inequality of Balas (1975) for $\text{conv}(X)$ as follows. For some $\hat{G} \subseteq G$, let $K = \bigcup_{g \in \hat{G}} J_g$, $\overline{K} = J \setminus K$, and $G_K = \left\{g \in G : j \in J_g \text{ for some } j \in K\right\}$. The set K is called a *GUB cover* of X if $\sum_{g \in G_{\overline{K}}} a_{j(g)} < b$. A GUB cover is called a *minimal GUB* *cover* of X if $\sum_{g\in G_{\overline{K}}} a_{j(g)} + \min_{g\in G_K}\left(a_{j(g)}\right) \geq b$. Accordingly, *a minimal GUB cover inequality* is written as

$$\sum_{j\in K} x_j \geq 1. \tag{6.1}$$

For some minimal GUB cover K, let $R = \left\{j' \in J \setminus K : a_{j'} \geq \max_{j\in K} a_j\right\}$. An extension of the minimal GUB cover, denoted by $E(K)$, is defined as $E(K) = K \cup \left(\bigcup_{g\in G_R} J_g\right)$ and a family of valid inequalities for $\mathrm{conv}(X)$ is defined as

$$\sum_{j\in E(K)} x_j \geq 1 + \left|G_R\right|. \tag{6.2}$$

If $R \neq \emptyset$, inequality (6.2) implies (i.e., dominates) (6.1); that is, if $R \neq \emptyset$, (6.2) is tighter than (6.1). Sherali and Lee (1995) defined another strengthening procedure for minimal covers as follows. If (K_1, K_2) is a partition of K (i.e., $K = K_1 \cup K_2$) with $K_2 \neq \emptyset$ such that

$$\sum_{g\in G_{K_2}} \max_{j\in J_g \cap K_2}(a_j) + \sum_{g\in G_{\overline{K}}} a_{j(g)} < b,$$

then inequality

$$\sum_{j\in K_1} x_j \geq 1, \tag{6.3}$$

is valid for $\mathrm{conv}(X)$ and dominates (6.1). Finally, given a minimal GUB cover K, Sherali and Lee (1995) developed a lifting procedure for (6.1), obtaining valid inequalities of the form

$$\sum_{j\in K} x_j + \sum_{j\in \overline{K}_-} \pi_j x_j + \sum_{j\in \overline{K}_+} \pi_j x_j \geq 1 + \sum_{j\in \overline{K}_+} \pi_j, \tag{6.4}$$

where $\overline{K}_+ = \left\{a_{j(g)} : g \in G_K\right\}$, $\overline{K}_- = J \setminus \left(K \cup \overline{K}_+\right)$, and π_j is the lifted coefficient of x_j.

The lifting procedure in Sherali and Lee (1995) computes lifted coefficients of the variables in each GUB set simultaneously. We now give an example from Sherali and Lee (1995) to demonstrate the valid inequalities that can be obtained using the procedures it presents.

Example 6.1.

$$X_{E1} = \left\{ \begin{array}{l} x \in \{0,1\}^8 : x_1 + 5x_2 + x_3 + 5x_4 + x_5 + 3x_6 + x_7 + 3x_8 \geq 9, \\ \qquad\qquad x_1 + x_2 \leq 1, \quad x_3 + x_4 \leq 1, \quad x_5 + x_6 \leq 1, \quad x_7 + x_8 \leq 1 \end{array} \right\}.$$

All possible minimal covers of form (6.1) are: $x_1 + x_2 + x_3 + x_4 \geq 1$;

$x_1 + x_2 + x_5 + x_6 \geq 1$;

$x_1 + x_2 + x_7 + x_8 \geq 1$;

$x_3 + x_4 + x_5 + x_6 \geq 1$;

$x_3 + x_4 + x_7 + x_8 \geq 1$.

All possible valid inequalities of form (6.2) are: $x_1 + x_2 + x_3 + x_4 + x_7 + x_8 \geq 2$;

$x_1 + x_2 + x_3 + x_4 + x_5 + x_6 \geq 2$.

The only possible valid inequality of form (6.3) is: $x_2 + x_4 \geq 1$.

Applying the lifting procedures of Sherali and Lee (1995), we obtain valid inequalities of the form (6.4):

$$x_1 + x_2 + x_3 + x_4 + \pi_5^1 x_5 + \pi_6^1 x_6 + \pi_7^1 x_7 + \pi_8^1 x_8 \geq 1 + \pi_6^1 + \pi_8^1$$

$$x_1 + x_2 + x_5 + x_6 + \pi_3^2 x_3 + \pi_7^2 x_7 + \pi_4^2 x_4 + \pi_8^2 x_8 \geq 1 + \pi_4^2 + \pi_8^2$$

$$x_1 + x_2 + x_7 + x_8 + \pi_3^3 x_3 + \pi_5^3 x_5 + \pi_4^3 x_4 + \pi_6^3 x_6 \geq 1 + \pi_4^3 + \pi_6^3$$

$$x_3 + x_4 + x_5 + x_6 + \pi_1^4 x_1 + \pi_7^4 x_7 + \pi_2^4 x_2 + \pi_8^4 x_8 \geq 1 + \pi_2^4 + \pi_8^4$$

$$x_3 + x_4 + x_7 + x_8 + \pi_1^5 x_1 + \pi_5^5 x_5 + \pi_2^5 x_2 + \pi_6^5 x_6 \geq 1 + \pi_2^5 + \pi_6^5 .$$

We show later in Example 6.3 that the valid inequalities that we propose differ from these.

6.2. Valid inequalities for $\mathbf{KPG}^{\geq}$

In this section, we derive a set of valid inequalities, called α-cover inequalities, for $\text{conv}(X)$. Assuming, without loss of generality, that $a_{j(1)} \geq a_{j(2)} \geq \ldots \geq a_{j(|G|)}$, we define

$$\alpha^* = \arg\min\left\{k \in \{1 \ldots |G|\} : \sum_{g=1}^{k} a_{j(g)} \geq b\right\} \tag{6.5}$$

and let $J^1(x) = \{j : x_j = 1\}$ be the index set of variables equal to 1 at feasible point $x \in X$.

Lemma 6.4. Given $J' \subseteq J$, $p = \min\left\{\sum_{j \in J'} x_j : x \in X\right\} \leq \alpha^*$.

Proof. Let $\hat{x}$ be a feasible solution with respect to X in which exactly α^* variables are fixed to 1 (i.e., $|J^1(\hat{x})| = \alpha^*$). By definition of α^*, $\min_{x \in X} |J^1(x)| = |J^1(\hat{x})| = \alpha^*$. Hence,

$$p = \min\left\{\sum_{j \in J'} x_j : x \in X\right\} = \min_{x \in X} |J^1(x) \cap J'| \leq |J^1(\hat{x}) \cap J'| \leq \alpha^* . \qquad \blacksquare$$

Definition 6.1. For each integer α such that $1 \leq \alpha \leq \alpha^*$, set $J^{\alpha} \subseteq J$ is an α*-cover*, if

$$|J^{\alpha} \cap J^1(x)| \geq \alpha \text{ for each } x \in X \text{ and} \tag{6.6}$$

$$\text{for each } j \in J^{\alpha}, \text{ an } x \in X \text{ exists such that } j \in J^1(x) \text{ and } |J^{\alpha} \cap J^1(x)| = \alpha . \tag{6.7}$$

Definition 6.1 justifies the following proposition.

Proposition 6.5. For any α-cover, $J^{\alpha} \subseteq J$,

$$\sum_{j \in J^{\alpha}} x_j \geq \alpha, \tag{6.8}$$

is a valid inequality for $\text{conv}(X)$.

Condition (6.6), which requires that J^{α} contain at least α variables from each $J^1(x)$ $x \in X$, assures that (6.8) is valid for $\text{conv}(X)$. By condition (6.7), for each $j \in J^{\alpha}$, a feasible point $\hat{x}$ exists such that $J^{\alpha} \setminus \{j\}$ contains the indices of at most $\alpha - 1$ of the variables that are fixed to 1 at $\hat{x}$. Hence, no $J^{\alpha} \setminus \{j\} \subset J^{\alpha}$ either satisfies (6.6) (on substituting $J^{\alpha} \setminus \{j\}$ for J^{α}) or yields an inequality that dominates (6.8) (Sherali and Glover 2008).

We call an inequality of form (6.8) an α-cover inequality. Example 6.2 demonstrates that α-cover inequalities may yield facets that differ from those that can be generated using the procedures of Sherali and Lee (1995).

Example 6.2. Consider polytope X_{E1} of Example 6.1. Note that $\alpha^* = 2$, since setting $x_2 = x_4 = 1$ gives $a_2 + a_4 \geq 9$. Given $J^{\alpha} = \{2, 4, 6, 8\}$ and $\alpha = \alpha^*$, we will show that J^{α} satisfies conditions (6.6) and (6.7), so that the corresponding inequality

$$x_2 + x_4 + x_6 + x_8 \geq 2 \tag{6.9}$$

is an α-cover inequality.

J^{α} satisfies (6.6), since $\min_{x \in X_{E1}} \left| J^{\alpha} \cap J^1(x) \right| = \min\{x_2 + x_4 + x_6 + x_8 : x \in X_{E1}\} = 2 = \alpha$.

We now show that J^{α} satisfies (6.7), by showing that for each $j \in J^{\alpha}$, a feasible solution x exists such that $j \in J^1(x)$ and $\left| J^{\alpha} \cap J^1(x) \right| = \alpha$.

Case 1. Consider point $x \in X_{E1}$ in which $x_2 = x_4 = 1$ and $x_j = 0 \;\; j \in \{1,3,5,6,7,8\}$ (i.e.,

$J^1(x) = \{2,4\}$). Thus, for $j \in \{2,4\}$, $j \in J^1(x)$ and $\left|J^\alpha \cap J^1(x)\right| = 2$.

Case 2. Consider $x \in X_{E1}$ in which $x_2 = x_3 = x_6 = 1$ and $x_j = 0 \;\; j \in \{1,4,5,7,8\}$ (i.e.,

$J^1(x) = \{2,3,6\}$). Thus, for $j = 6$, $j \in J^1(x)$ and $\left|J^\alpha \cap J^1(x)\right| = 2$.

Case 3. Consider $x \in X_{E1}$ in which $x_2 = x_3 = x_8 = 1$ and $x_j = 0 \;\; j \in \{1,4,5,6,7\}$ (i.e.,

$J^1(x) = \{2,3,8\}$). Thus, for $j = 8$, $j \in J^1(x)$ and $\left|J^\alpha \cap J^1(x)\right| = 2$.

By cases 1-3, for each $j \in J^\alpha$, an $x \in X_{E1}$ exists such that $j \in J^1(x)$ and $\left|J^\alpha \cap J^1(x)\right| = \alpha$.

In fact, (6.9) is a facet of X_{E1}, since array

$$\mathbf{M} = \begin{array}{c} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{array} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

gives eight linearly independent points in X_{E1} for which (6.9) holds at equality. Further, (6.9) cannot be generated using the procedures of Sherali and Lee (1995) (see Example 6.1 in which we list all valid inequalities (facets) that can be generated using the procedures of Sherali and Lee (1995)).

Note that (6.8) generalizes the GUB cover inequalities described in Sherali and Lee (1995). The right-hand-side of any GUB cover inequality is 1 and either all variables associated with a GUB are in a GUB cover or in the complement of the GUB cover. The α-cover inequality generalizes the GUB cover inequality, because the right-hand-side of an α-cover inequality (6.8) can be larger than 1 and variables associated with a GUB can be partitioned into *two* subsets: one is included in the α-cover; and the other, in the complement of the α-cover (i.e., $J \setminus J^{\alpha}$).

6.3. Generating α-cover inequalities

This section designs a polynomial-time procedure for generating an α-cover inequality. Later, in Section 6.7, we use this procedure to find the most violated α-cover inequality.

Notation. We introduce the following notation. Given an index subset $H \subseteq J$,

$\overline{H} = J \setminus H$ is the complement of H; and the subset of variable indices that are common to both J_g and H ($\overline{H}$) is $J_g^H = H \cap J_g$ $\left(J_g^{\overline{H}} = \overline{H} \cap J_g\right)$;

G_H ($G_{\overline{H}}$) is the index set of GUBs, each of which is associated with one or more variables in H ($\overline{H}$).

Note that each GUB can be an element in G_H **and/or** $G_{\overline{H}}$.

$G_{H\overline{H}}$ is the index set of GUBs that have one or more elements in both G_H and $G_{\overline{H}}$.

N_H ($N_{\overline{H}}$) is the index set of GUBs with no associated variables in H $\left(\overline{H}\right)$; that is $g \in N_H$ if $J_g^H = \varnothing$ $\left(g \in N_{\overline{H}} \text{ if } J_g^{\overline{H}} = \varnothing\right)$.

Proofs of the propositions and lemmas that follow deal with the variables with the largest coefficients in set $J_g^H \left(J_g^{\overline{H}}\right)$, so we define notation to denote such variables:

for each $g \in G_H$, $j_H(g)$ is chosen such that $j_H(g) \in \arg\max\left\{a_j : j \in J_g^H\right\}$; (6.10)

for each $g \in G_{\overline{H}}$, $j_{\overline{H}}(g)$ is chosen such that $j_{\overline{H}} \in \arg\max\left\{a_j : j \in J_g^{\overline{H}}\right\}$. (6.11)

For each $g \in G_{H\overline{H}}$, if $x_{j_{\overline{H}}(g)} = 1$ and is subsequently replaced by $x_{j_H(g)} = 1$, the left-hand-side of the $\text{KPG}^{\geq}$ knapsack constraint changes in value by

$$\breve{a}_{j_H(g)} = a_{j_H(g)} - a_{j_{\overline{H}}(g)}. \tag{6.12a}$$

Analogously, since $G_H \bigcap G_{\overline{H}} = \varnothing$, for each $g \in G_H$, we define

$$\breve{a}_{j_H(g)} = a_{j(g)}. \tag{6.12b}$$

Algorithm **COVER(*D*, *α*)** takes a non-empty index set $D \subseteq J$ and parameter α as inputs, and produces an α-cover. We assume that $\min\left\{\sum_{j\in D} x_j : x \in X\right\} \geq \alpha$; otherwise, variables in D do not yield an α-cover inequality that is valid for $\text{conv}(X)$.

Each iteration of **COVER(*D*, *α*)** requires a problem of the following form to be solved for a given $H \subseteq D$:

$$Z_H^* = \min\left\{\sum_{j\in H} x_j : x \in X\right\}. \tag{6.13}$$

Proposition 6.6 prescribes an optimal solution to Problem (6.13). It states that, if starting with $x_j = 0$ for all $j \in J$, then fixing the variable with the largest a_j value in each $g \in G_{\overline{H}}$ to 1 (i.e., $x_{j_{\overline{H}}(g)} = 1$) satisfies the knapsack constraint in (6.13), $Z_H^* = 0$; otherwise, we must increase Z_H^* by fixing one or more variables in H to 1. The minimum number of variables in H that must be fixed to 1 can be found by successively fixing the variable $x_{j_H(g)}$ with the largest $\breve{a}_{j_H(g)}$ value to 1 and, for $g \in G_{\overline{H}}$, the corresponding $x_{j_{\overline{H}}(g)} = 0$ until feasibility is achieved.

Proposition 6.6. Let $t = |G_H|$. Sort the indices in G_H in non-increasing order of their $\breve{a}_{j_H(g)}$ values and re-number, so that $\breve{a}_{j_H(1)} \geq \breve{a}_{j_H(2)} \geq ... \geq \breve{a}_{j_H(t)}$. Then, with $x_{j_{\overline{H}}(g)} = 1$ for $g \in G_{\overline{H}}$,

$$Z_H^* = \arg\min\left\{k \in \{1...t\} : \sum_{g=1}^{k} \breve{a}_{j(g)}^H \geq b - \sum_{g \in G_{\overline{H}}} a_{j_{\overline{H}}(g)}\right\}. \tag{6.14}$$

Proof. We recast Problem (6.13) as a knapsack problem without GUBs by considering two cases. In Case 1, we show that an optimal solution to (6.13) exists in which, for each $g \in G_{\overline{H}}$, $x_j = 0$ for all $j \in J_g^{\overline{H}}$, except for variable $x_{j_{\overline{H}}(g)}$. Similarly, in Case 2, we show that for each $g \in G_H$, $x_j = 0$ for all $j \in J_g^H$, except for variable $x_{j_H(g)}$. We arbitrarily break ties while selecting $x_{j_H(g)}$ and $x_{j_{\overline{H}}(g)}$.

Case 1. Consider $g \in G_{\overline{H}}$. The objective function coefficient of each x_j $j \in J_g^{\overline{H}}$ is 0 in (6.13) and $a_{j_{\overline{H}}(g)} \geq a_j$ for each $j \in J_g^{\overline{H}}$ by (6.11). Therefore, we prefer to fix $x_{j_{\overline{H}}(g)}$ to 1 instead of x_j $j \in J_g^{\overline{H}} \setminus \{j_{\overline{H}}(g)\}$. Thus, an optimal solution to (6.13) exists in which $\sum_{j \in J_g^H} x_j + x_{j_{\overline{H}}(g)} = 1$ for each $g \in G_{\overline{H}}$ and other associated variables are fixed to 0; i.e., $x_j = 0$ for $j \in J_g^{\overline{H}} \setminus \{j_{\overline{H}}(g)\}$.

Case 2. Consider $g \in G_H$. The objective function coefficient for each x_j $j \in H$ is 1 in (6.13) and $a_{j_H(g)} \geq a_j$ for each $j \in J_g^H$ by (6.10). Hence, we prefer to fix $x_{j_H(g)}$

to 1 instead of $x_j \quad j \in J_g^H \setminus \{j_H(g)\}$. Thus, in all cases, an optimal solution to (6.13) exists such that

$$x_{j_H(g)} + x_{j_{\overline{H}}(g)} = 1 \text{ for } g \in G_{H\overline{H}} \qquad \text{and} \qquad x_{j_H(g)} \le 1 \text{ for } g \in N_H .$$

After fixing $x_j = 0$ for $j \in J \setminus \bigcup_{g \in G_H} \{j_H(g)\} \setminus \bigcup_{g \in G_{\overline{H}}} \{j_{\overline{H}}(g)\}$, and $x_{j(g)} = 1$ for $g \in N_H$, and replacing each GUB $g \in G_{H\overline{H}}$ with an equality constraint, (6.13) becomes

$$Z_H^* = \min \sum_{j \in H} x_j$$

$$\text{s.t. } \sum_{g \in G_H} a_{j_H(g)} x_{j_H(g)} + \sum_{g \in G_{H\overline{H}}} a_{j_{\overline{H}}(g)} x_{j_{\overline{H}}(g)} \ge b - \sum_{g \in N_H} a_{j(g)} \tag{6.15}$$

$$x_{j_H(g)} + x_{j_{\overline{H}}(g)} = 1 \qquad g \in G_{H\overline{H}} \tag{6.16}$$

$$x_{j_H(g)} \in \{0,1\} \qquad g \in G_H$$

$$x_{j_{\overline{H}}(g)} \in \{0,1\} \qquad g \in G_{H\overline{H}} .$$

We use (6.16) to replace each $x_{j_{\overline{H}}(g)}$ in (6.15) with $(1 - x_{j_H(g)})$. Then, (6.15) becomes

$$\sum_{g \in G_H} a_{j_H(g)} x_{j_H(g)} - \sum_{g \in G_{H\overline{H}}} a_{j_{\overline{H}}(g)} x_{j_H(g)} \ge b - \sum_{g \in N_H} a_{j(g)} - \sum_{g \in G_{H\overline{H}}} a_{j_{\overline{H}}(g)} . \tag{6.15a}$$

Considering $G_H = N_{\overline{H}} \cup G_{H\overline{H}}$ and $G_{\overline{H}} = N_H \cup G_{H\overline{H}}$, (6.15a) can re-expressed as

$$\sum_{g \in N_{\overline{H}}} a_{j_H(g)} x_{j_H(g)} + \sum_{g \in G_{H\overline{H}}} a_{j_H(g)} x_{j_H(g)} - \sum_{g \in G_{H\overline{H}}} a_{j_{\overline{H}}(g)} x_{j_H(g)} \ge b - \sum_{g \in G_{\overline{H}}} a_{j_{\overline{H}}(g)} . \tag{6.15b}$$

Replacing (6.15) with (6.15b), Problem (6.13) becomes

$$Z_H^* = \min \sum_{j \in H} x_j$$

$$\begin{aligned} \text{s.t. } & \sum_{g \in N_{\overline{H}}} a_{j_H(g)} x_{j_H(g)} + \sum_{g \in G_{H\overline{H}}} (a_{j_H(g)} - a_{j_{\overline{H}}(g)}) x_{j_H(g)} \ge b - \sum_{g \in G_{\overline{H}}} a_{j_{\overline{H}}(g)} \\ & x_{j_H(g)} \in \{0,1\} \qquad g \in G_H . \end{aligned} \tag{6.17}$$

After invoking (6.12) to set $\breve{a}_{j_H(g)} = a_{j(g)}$ for $g \in N_{\overline{H}}$ in the first summation in (6.17), $\breve{a}_{j_H(g)} = a_{j_H(g)} - a_{j_{\overline{H}}(g)}$ for $g \in G_{H\overline{H}}$ in the second summation in (6.17), and recognizing that $G_H = N_{\overline{H}} \cup G_{H\overline{H}}$, Problem (6.13) becomes

$$Z_H^* = \min\left\{\sum_{j\in H} x_j : \sum_{g\in G_H} \breve{a}_{j_H(g)} x_{j_H(g)} \geq b - \sum_{g\in G_{\overline{H}}} a_{j_{\overline{H}}(g)}; x_{j_H(g)} \in \{0,1\}\ g \in G_H \right\}.$$

Now, observe that if GUBs in G_H are sorted in non-increasing order of their $\breve{a}_{j_H(g)}$ values and re-numbered, Z_H^* is given by the smallest integer $k \leq t$ for which

$\sum_{g=1}^{k} \breve{a}_{j_H(g)} \geq b - \sum_{g\in G_{\overline{H}}} a_{j_{\overline{H}}(g)}$, establishing (6.14). ■

Proposition 6.6 has practical significance. It implies that if the GUBs in G_H are sorted in non-increasing order of their $\breve{a}_{j_H(g)}$ values, Z_H^* can be found in $O(|G|)$time. Algorithm **COVER(*D*, *α*)** begins with $H = D$ such that GUBs in G_D (i.e., G_H) are sorted in non-increasing order of their $\breve{a}_{j_D(g)}$ (i.e., $\breve{a}_{j_H(g)}$) values. At each iteration, it fixes a different $x_{\hat{j}}$ $\hat{j} \in H$ to 1 and then determines if $Z_H^*(\hat{j}) > \alpha$ or not, where

$$Z_H^*(\hat{j}) = \min\left\{\sum_{j\in H} x_j : x \in X, x_{\hat{j}} = 1\right\}. \tag{6.18}$$

Let $\hat{g}$ be such that $\hat{j} \in J_{\hat{g}}$. Since sorting the GUBs each time one variable is fixed may be time consuming, Corollary 6.7 demonstrates how to determine if $Z_H^*(\hat{j}) > \alpha$ or not in constant time.

Corollary 6.7. Consider Problem (6.18). Assume that GUBs are sorted in G_H in non-increasing order of their $\breve{a}_{j_H(g)}$ values and $a_{j_{\overline{H}}(g)}$ is set to 0 if $g \in N_{\overline{H}}$. Let

$$G_H^+ = \{1, \ldots, \alpha\}$$

and $$b(\hat{j}) = \begin{cases} w_H - a_{j_H(\hat{g})} + a_{\hat{j}} & \text{if } \hat{g} \le \alpha \\ w_H - a_{j_H(\alpha)} + a_{j_{\bar{H}}(\alpha)} - a_{j_{\bar{H}}(\hat{g})} + a_{\hat{j}} & \text{if } \hat{g} > \alpha \end{cases},$$

where $$w_H = \sum_{g \in G_H^+} a_{j_H(g)} + \sum_{g \in G_{\bar{H}} \setminus G_H^+} a_{j_{\bar{H}}(g)} . \tag{6.19}$$

If $b > b(\hat{j})$, then $Z_H^*(\hat{j}) > \alpha$.

Proof. Let $\hat{x}$ be such that $\hat{x}_{j_H(g)} = 1$ for $g \in G_H^+$, $\hat{x}_{j_{\bar{H}}(g)} = 1$ for $g \in G_H \setminus G_H^+$, and $\hat{x}_j = 0$ for each remaining variable. The sum of coefficient values (i.e., a_j) associated with $\hat{x}$ is given by w_H (6.19). Now, suppose that a solution x exists in which $x_{\hat{j}} = 1$ and exactly α variables from H are fixed to 1. Then, by Proposition 6.6, an x exists in which exactly $(\alpha - 1)$ variables $x_{j_H(g)}$ with the largest $\breve{a}_{j_H(g)}$ values are fixed to 1. The sum of coefficient values associated with x can be calculated based on two cases: $\hat{g} \in G_H^+$ (i.e., $\hat{g} \le \alpha$) and $g \in G_H \setminus G_H^+$ (i.e., $\hat{g} > \alpha$).

Case 1. If $\hat{g} \le \alpha$, then x can be obtained from $\hat{x}$ by replacing $\hat{x}_{j_H(\hat{g})} = 1$ with $x_{\hat{j}} = 1$.

Therefore, $\sum_{j \in J^1(x)} a_j = w_H - a_{j_H(g)} + a_{\hat{j}}$.

Case 2. If $\hat{g} > \alpha$, then x can obtained from $\hat{x}$ by replacing $\hat{x}_{j_H(\alpha)} = 1$ with $\hat{x}_{j_{\bar{H}}(\alpha)} = 1$;

and $\hat{x}_{j_{\bar{H}}(\hat{g})} = 1$ with $x_{\hat{j}} = 1$. Therefore,

$$\sum_{j \in J^1(x)} a_j = w_H - a_{j_H(\alpha)} + a_{j_{\bar{H}}(\alpha)} - a_{j_{\bar{H}}(\hat{g})} + a_{\hat{j}} .$$

By Cases 1 and 2, if $b \le b(\hat{j})$, x is feasible and $Z_H^*(\hat{j}) \le \alpha$; otherwise $Z_H^*(\hat{j}) > \alpha$. ■

After giving a statement of **COVER(*D*, *α*)** in pseudo code, we give an intuitive description of each step.

COVER(D, α):

Input : Set D and parameter α
Output: Set J^{α}

(1a) $H \leftarrow D$, $D' \leftarrow \varnothing$ and compute w_H using (6.19).
(2a) **for** each $g \in G_H$ and $j \in J_g^H$ **do**
(3a) **if** ($a_j \le a_{j_{\bar{H}}(g)}$) **then** $H \leftarrow H \setminus \{j\}$ and $D' \leftarrow D' \cup \{j\}$
(4a) **while** $D \neq D'$
(5a) $\hat{j} \leftarrow \arg\min_{j \in H \setminus D'} \{a_j\}$, select $\hat{g}$ such that $\hat{j} \in J_{\hat{g}}$, $D' \leftarrow D' \cup \{\hat{j}\}$ and
(6a) **if** $(b > b(\hat{j}))$ **then**
(7a) $H \leftarrow H \setminus \{\hat{j}\}$, $J_{\hat{g}}^H \leftarrow J_{\hat{g}}^H \setminus \{\hat{j}\}$, $\breve{a}_{prev} \leftarrow \breve{a}_{j_H(\hat{g})}$, and $\breve{a}_{j_H(\hat{g})} \leftarrow a_{j(\hat{g})} - a_{\hat{j}}$
(8a) **if** ($\hat{g} < |G_H|$ and $\breve{a}_{j_H(\hat{g})} < \breve{a}_{j_H(\hat{g}+1)}$) **then**
(9a) $\tilde{g} \leftarrow \hat{g} + 1$, and $J_{prev} \leftarrow J_{\hat{g}}^H$
(10a) **while** ($\tilde{g} + 1 \le |G_H|$ and $\breve{a}_{j_H(\hat{g})} < \breve{a}_{j_H(\tilde{g}+1)}$) **do** $\tilde{g} \leftarrow \tilde{g} + 1$
(11a) **for** $g \leftarrow \hat{g} + 1$ **to** $\tilde{g}$ **do**
(12a) $J_{g-1}^H \leftarrow J_g^H$, $\breve{a}_{j_H(g-1)} \leftarrow \breve{a}_{j_H(g)}$ and $a_{j_{\bar{H}}(g-1)} \leftarrow a_{j_{\bar{H}}(g)}$
(13a) $J_{\tilde{g}}^H \leftarrow J_{prev}$, $\breve{a}_{j_H(\tilde{g})} \leftarrow a_{j(\hat{g})} - a_{\hat{j}}$ and $a_{j_{\bar{H}}(\tilde{g})} \leftarrow a_{\hat{j}}$
(14a) **if** ($\hat{g} > \alpha$) **then** $w_H \leftarrow w_H + \breve{a}_{prev} - \breve{a}_{j_H(\tilde{g})}$
(15a) **else if** ($\hat{g} \le \alpha$ and $\tilde{g} > \alpha$) **then** $w_H \leftarrow w_H - a_{j(\tilde{g})} + a_{\hat{j}} + \breve{a}_{j_H(\alpha)}$
(16a) **else** $D' \leftarrow D' \cup J_{\hat{g}}^H$
(17a) $J^{\alpha} \leftarrow H$.

Step (1a) initializes the algorithm with $H = D$ and $D' = \varnothing$, where D' is the index set of variables considered during previous iterations. Lemma 6.8 establishes that, if $j \in J_g$ is not in J^{α}, then J^{α} does not contain the index of any variable x_i $i \in J_g$ whose coefficient a_i is less-than-or-equal-to coefficient a_j of x_j. Hence, if $a_j \le a_{j_{\bar{H}}(g)}$ for $j \in J_g^H$, we remove j from H (Steps (2a)-(3a)). If $a_{j_{\bar{H}}(g)} = a_{j(g)}$, we remove all $j \in J_g^H$ from H. Therefore, $j_H(g) = j(g)$ for the GUBs associated with the remaining variables in H.

Lemma 6.8. Given $\hat{g}$, consider variable $x_{\hat{j}}$ such that $\hat{j} \in J_{\hat{g}}$.

(i) If $\hat{j} \in J^{\alpha}$, then J^{α} contains all indices in $R_{\geq \hat{j}} = \{i \in J_{\hat{g}} : a_i \geq a_{\hat{j}}\}$; i.e., $J^{\alpha} \cap R_{\geq \hat{j}} = R_{\geq \hat{j}}$.

(ii) If $\hat{j} \notin J^{\alpha}$, then J^{α} does not contain any index in $R_{\leq \hat{j}} = \{i \in J_{\hat{g}} : a_i \leq a_{\hat{j}}\}$; i.e., $J^{\alpha} \cap R_{\leq \hat{j}} = \varnothing$.

Proof. (i) Observe variable $x_{j'}$ such that $j' \in J_{\hat{g}} \setminus J^{\alpha}$. By condition (6.6), each $x \in X$ in which $x_{j'} = 1$ requires at least α additional variables from J^{α} to be fixed to 1. Hence, no $H_1 \subset J^{\alpha} \setminus J_{\hat{g}}$ exists such that $|H_1| = \alpha - 1$; fixing $x_{j'} = 1$, $x_j = 1$ for $j \in H_1$, $x_{j_{\overline{H}}(g)} = 1$ for $g \in G_H \setminus G_{H_1}$, and $x_j = 0$ for each remaining variable gives a feasible solution with respect to X. Thus, for any $H_1 \subset J^{\alpha} \setminus J_{\hat{g}}$ such that $|H_1| = \alpha - 1$,

$$\sum_{j \in H_1} a_j + a_{j'} + \sum_{g \in G_{\overline{H}_1} \setminus \{\hat{g}\}} a_{j_{\overline{H}_1}(g)} < b. \tag{6.20}$$

By way of contradiction, suppose that, for $\hat{j} \in J^{\alpha}$, $j' \notin J^{\alpha}$ exists such that $j' \in R_{\geq \hat{j}}$. Condition (6.7) stipulates that an $\hat{x} \in X$ exists in which $\hat{x}_j = 1$ and exactly $\alpha - 1$ variables from $J^{\alpha} \setminus J_{\hat{g}}$ are fixed to 1. Define $H_2 = J^1(\hat{x}) \cap (J^{\alpha} \setminus J_{\hat{g}})$. Since exactly $\alpha - 1$ variables from $J^{\alpha} \setminus J_{\hat{g}}$ are fixed to 1, $|H_2| = \alpha - 1$. Furthermore, since $\hat{x} \in X$,

$$\sum_{j \in H_2} a_j + a_{\hat{j}} + \sum_{g \in G_{\overline{H}_2} \setminus \{\hat{g}\}} a_{j_{\overline{H}_2}(g)} \geq b, \tag{6.21}$$

Since $a_{j'} \geq a_{\hat{j}}$, we obtain $\sum_{j\in H_2} a_j + a_{j'} + \sum_{g\in G_{\overline{H}_2}\setminus\{\hat{g}\}} a_{j_{\overline{H}_2}(g)} \geq b$ by replacing $a_{\hat{j}}$ with $a_{j'}$ in (6.21); contradicting (6.20). Hence, $J^{\alpha} \cap R_{\geq \hat{j}} = R_{\geq \hat{j}}$.

(ii) A similar argument can be used to prove that $J^{\alpha} \cap R_{\leq \hat{j}} = \varnothing$. ■

At each iteration (Steps (5a)-(16a)), we choose $x_{\hat{j}}$ such that $\hat{j} \in \arg\min_{j\in H\setminus D'}\{a_j\}$ and $\hat{g}$ such that $\hat{j} \in J_{\hat{g}}$ (Step 5a). We remove $\hat{j}$ from H if we need to fix at least α variables from $H \setminus \{\hat{j}\}$ to 1 in order to satisfy $\sum_{g\in G\setminus\{\hat{g}\}}\sum_{j\in J_g} a_j x_j \geq b - a_{\hat{j}}$ (Steps (6a)-(7a)); i.e., $Z_H^*(\hat{j}) > \alpha$.

In order to determine if $Z_H^*(\hat{j}) > \alpha$ or not in constant time at each iteration, we must keep GUBs in G_H sorted (and re-numbered) in non-increasing order of their $\breve{a}_{j_H(g)}$ values. If $\hat{g} = |G_H|$, the sorted order of the GUBs in G_H does not change by removing $\hat{j}$ from H. However, if $\hat{g} < |G_H|$, after removing $\hat{j}$ from H, the value of $\breve{a}_{j_H(\hat{g})}$ must be reduced to $a_{j_H(\hat{g})} - a_{\hat{j}}$ (Step (7a)). Therefore, the order of GUBs in G_H may change. In this case, we can utilize the following scheme to update the order of GUBs in G_H:

if $a_{j_H(\hat{g})} - a_{\hat{j}} > \breve{a}_{j_H(|G_H|)}$, let $\tilde{g} \in \arg\max\{g \in G_H : a_{j_H(\hat{g})} - a_{\hat{j}} < \breve{a}_{j_H(g)}, g \neq \hat{g}\}$;

otherwise , let $\tilde{g} = |G_H|$;

decrease the indices of GUBs in $[\hat{g}+1, \tilde{g}]$ by 1; change the previous index of GUB $\hat{g}$ to $\tilde{g}$ and update the value of w_H and the attributes of relevant GUBs: J_g^H, $\breve{a}_{j_H(g)}$, and $a_{j_{\overline{H}}(g)}$, accordingly (Steps (8a)-(15a)).

Lemma 6.8 asserts that if $j \in J_g$ is in J^α, then J^α contains the index of each variable x_i $i \in J_g$ whose coefficient (i.e., a_i) is greater-than-or-equal-to coefficient a_j of x_j. Hence, whenever we do not remove $\hat{j}$ from H, we keep all $j \in J_g$ with $a_j \geq a_{\hat{j}}$ in H and we do not consider them in subsequent iterations (Step (16a)). We repeat Steps (5a)-(16a) until all variables have been processed (i.e., $D = D'$).

The set H obtained at Step (17a) of **COVER(*D*, *α*)** satisfies conditions (6.6) and (6.7), so that it is an α-cover of X. The set H satisfies (6.6), because Step (6a) removes index $\hat{j} \in J_{\hat{g}}^H$ from H only if fixing $x_{\hat{j}} = 1$ requires at least α additional variables from J^α to be fixed to 1 (i.e., if $Z_H^*\left(\hat{j}\right) > \alpha$). The set H satisfies (6.7), since we check each variable in D and remove each $\hat{j}$ from H if $\left|\left(H \setminus \left\{\hat{j}\right\}\right) \cap J^1(x)\right| \geq \alpha$ for all $x \in X$ (Steps (6a)-(7a)).

We use a numerical example from Sherali and Lee (1995) to demonstrate **COVER(*D*, *α*)** in application.

Example 6.3 (Sherali and Lee 1995).

$$X_{E2} = \left\{ \begin{array}{l} x \in B^9 : x_1 + x_2 + 3x_3 + x_4 + x_5 + 2x_6 + x_7 + x_8 + 2x_9 \geq 4, \\ \qquad\qquad x_1 + x_2 + x_3 \leq 1, \quad x_4 + x_5 + x_6 \leq 1, \quad x_7 + x_8 + x_9 \leq 1 \end{array} \right\}.$$

COVER(*D*=$\{1,\ldots,9\}$**, *α* =2)** requires the following five iterations:

(1a) $H = \{1 \ldots 9\}$, $D' = \varnothing$

First iteration:	**Second iteration:**
(5a) $\hat{j} = 1$, $\hat{g} = 1$, $D' = \{1\}$	(5a) $\hat{j} = 2$, $\hat{g} = 1$, $D' = \{1,2\}$
(7a) $b(1) = 3 < b$	(7a) $b(2) = 3 < b$
(8a) $H = \{2,\ldots,9\}$	(8a) $H = \{3,\ldots,9\}$

Third iteration:	Fourth iteration:	Fifth iteration:
(5a) $\hat{j}=4$, $\hat{g}=2$	(5a) $\hat{j}=7$, $\hat{g}=3$,	(5a) $\hat{j}=3$, $\hat{g}=1$,
$D'=\{1,2,4\}$	$D'=\{1,2,4,5,6,7\}$	$D'=\{1,\ldots,9\}$
(7a) $b(4)=4\geq b$	(7a) $b(7)=4\geq b$	(7a) $b(3)=5\geq b$
(15a) $D'=\{1,2,4,5,6\}$	(15a) $D'=\{1,2,4,\ldots,9\}$	(15a) $D'=\{1,\ldots,9\}$
Note: $H=\{3,\ldots,9\}$	Note: $H=\{3,\ldots,9\}$	Note: $H=\{3,\ldots,9\}$

Since $D=D'$, STOP. $J^{\alpha}=H=\{3,\ldots,9\}$.

Note that Sherali and Lee (1995) show that $x_3+x_4+x_5+x_6+x_7+x_8+x_9\geq 2$ is a facet of X_{E2}.

Proposition 6.9. Algorithm **COVER(*D*, *α*)** is of complexity $O(|G|+|D|^2)$.

Proof. Step (1a) requires $O(|D|+|G|)$ time. Together, Steps (2a)-(3a) require $O(|D|)$ time. Each iteration (i.e., Steps (5a)-(16a)), requires $O(|D|)$ time, (Steps (6a)-(8a) and Steps (14a)-(15a) each require constant time; Steps (5a), (9a), (10a), and (16a) each - and Steps (11a)-(13a) collectively - require $O(|D|)$ time). Since Step (4a) requires repeating Steps (5a)-(16a) in $O(|D|)$ times, Steps (4a)-(16a) collectively require $O(|D|^2)$ time. Step (17a) requires $O(|D|)$ time. Thus, the overall time complexity of **COVER(*D*, *α*)** is $O(|G|+3|D|+|D|^2)$, which reduces to $O(|G|+|D|^2)$. ■

Remark 6.1. We assume that Algorithm **COVER(*D*, *α*)** begins with $H=D$ whose associated GUBs G_H (i.e., G_D) are ordered in non-increasing order of their $\breve{a}_{j_H(g)}$ values. Consider the case in which GUBs in G_H are not sorted in order. By including a new step before Step (4a), we can order them in $O(|D|\log|D|)$ time (Cormen et al.

1990). Therefore, even if **COVER(D, α)** begins with an un-ordered G_H, it requires $O(|G|+|D|^2)$ time. ■

6.4. Non-dominated inequalities for $\mathbf{KPG}^{\geq}$

In this section we present a polynomial-time procedure to strengthen an α-cover. Consider a pair of inequalities for non-empty sets $J', J'' \subseteq J$,

$$\sum_{J'} x_j \geq \alpha' \tag{6.22}$$

$$\sum_{J''} x_j \geq \alpha''. \tag{6.23}$$

Glover and Sherali (2008) say that (6.22) *dominates* (6.23) if it implies (6.23). Then, they assert that (6.22) dominates (6.23) over the unit hypercube (i.e., $\{x : 0 \leq x \leq 1\}$) if either

$$J' \subseteq J'' \text{ and } \alpha' \geq \alpha'' \text{ (with at least one relation strict),} \tag{6.24}$$

or $$J' = J'' \cup \{j\} \text{ for some } j \in J \setminus J'' \text{ and } \alpha' = \alpha'' + 1. \tag{6.25}$$

Moreover, they say that, for a given J' and α', inequality (6.22) is *non-dominated* if $\alpha' \geq 1$ and if there does not exist another valid inequality that dominates it.

Let (6.22) be a non-dominated, valid inequality for $\text{conv}(X)$. Since (6.22) is valid for $\text{conv}(X)$, it satisfies condition (6.6). Furthermore, (6.22) satisfies (6.7) by condition (6.24). Since non-dominated inequality (6.22) satisfies (6.6) and (6.7), it is an α'-cover inequality by Definition 6.1. Proposition 6.11 shows that an α-cover inequality is non-dominated if a simple condition is satisfied. We first introduce the following notation.

Notation. To facilitate presentation, we simplify the notation used in Section 6.3 to denote the case in which $H = J^{\alpha}$. To avoid superscripts on subscripts, we use $\alpha\,(\overline{\alpha}\,)$ instead of $J^{\alpha}\,(J^{\overline{\alpha}}\,)$: $G_{\alpha} := G_{J^{\alpha}}$ and $N_{\alpha} := N_{J^{\alpha}}$ for $H = J^{\alpha}$;

$G_{\overline{\alpha}} := G_{\bar{J}^{\alpha}}$ and $N_{\overline{\alpha}} := N_{\bar{J}^{\alpha}}$ for $\overline{H} = \bar{J}^{\alpha}$;

$G_{\alpha\overline{\alpha}} := G_{J^{\alpha}\bar{J}^{\alpha}}$

$J_{g}^{\alpha} := J_{g}^{J^{\alpha}} = J^{\alpha} \bigcap J_{g}$ and $\bar{J}_{g}^{\alpha} := \bar{J}_{g}^{\bar{J}^{\alpha}} = \bar{J}^{\alpha} \bigcap J_{g}$ for $g \in G$.

We know by Lemma 6.8 that $a_{j_H(g)} = a_{j(g)}$ for $H = J^{\alpha}$.

We also eliminate subscripts $\overline{H}$ (i.e., $\bar{J}^{\alpha}$) and H (i.e., J^{α}) on $j_{\overline{H}}(g)$ and $a_{j_H(g)}$:

$\bar{j}(g) := j_{\overline{H}}(g)$ $\left(\text{i.e., } \bar{j}(g) \in \arg\max\{a_j : j \in \bar{J}_{g}^{\alpha}\}\right)$ for $g \in G_{\overline{\alpha}}$;

$$\breve{a}_{j(g)} = \begin{cases} a_{j(g)} & \text{for } g \in N_{\overline{\alpha}} \\ a_{j(g)} - a_{\bar{j}(g)} & \text{for } g \in G_{\alpha\overline{\alpha}} \end{cases}.$$

Let g_k be the index of the GUB in G_{α} with the k^{th} largest $\breve{a}_{j(g)}$ (ties are broken arbitrarily), i.e., $\breve{a}_{j(g_1)} \geq ... \geq \breve{a}_{j(g_k)} \geq ... \geq \breve{a}_{j(g_t)}$, where $t = |G_{\alpha}|$.

Finally, we define new notation that we use in Sections 6.4-6.6:

Let $G_{\alpha}^{+} = \{g_1, ..., g_{\alpha}\}$ be the index set of GUBs with the α largest $\breve{a}_{j(g)}$ values; and

$G_{\alpha}^{-} = G_{\alpha} \setminus G_{\alpha}^{+}\,(G_{\overline{\alpha}}^{-} = G_{\overline{\alpha}} \setminus G_{\alpha}^{+})$ be the subset of $G_{\alpha}\,(G_{\overline{\alpha}})$ that does not contain any index in G_{α}^{+}. By the definition of α-cover and Proposition 6.6, setting $x_{j(g)} = 1$ for $g \in G_{\alpha}^{+}$, $x_{\bar{j}(g)} = 1$ for $g \in G_{\overline{\alpha}}^{-}$, and $x_j = 0$ for each remaining variable gives a feasible point of X (i.e., $\text{conv}(X)$). We denote this point using

$$\delta_0 = \sum_{g \in G_\alpha^+} e_{j(g)} + \sum_{g \in G_{\bar{\alpha}}^-} e_{\bar{j}(g)} ,$$

where e_j is the unit vector that has 1 in the row corresponding to variable x_j and 0 in each other row. However, no variable associated with $g \in N_{\bar{\alpha}}$ is in $J^{\bar{\alpha}}$; therefore, we assume that $e_{\bar{j}(g)} = \mathbf{0}$ for $g \in N_{\bar{\alpha}}$. We also use w to denote the summation of the knapsack coefficients (i.e., a_j) corresponding to the variables in $J^1(\delta_0)$; i.e.,

$$w = \sum_{g \in G_\alpha^+} a_{j(g)} + \sum_{g \in G_{\bar{\alpha}}^-} a_{\bar{j}(g)} \quad . \tag{6.26}$$

By Definition 6.1, for each $j \in J^\alpha$, an $x \in X$ exists such that $j \in J^1(x)$ and $\left|J^\alpha \cap J^1(x)\right| = \alpha$. We now present Lemma 6.10, which shows how such an x can be obtained from δ_0 for each $j \in J_g^\alpha$ based on: $g \in G_\alpha^+$ and $g \in G_\alpha^-$. We use each point defined in the subsequent propositions.

Lemma 6.10. (i) For $g \in G_\alpha^+$ and $j \in J_g^\alpha$, the point $\hat{\delta} = \delta_0 - e_{j(g)} + e_j$, which is obtained by replacing $x_{j(g)} = 1$ in δ_0 with $x_j = 1$, is a feasible point of X.

(ii) For $g \in G_\alpha^-$ and $j \in J_g^\alpha$, the point $\hat{\delta} = \delta_0 - e_{j(g_\alpha)} + e_{\bar{j}(g_\alpha)} - e_{\bar{j}(g)} + e_j$, which is obtained by replacing $x_{j(g_\alpha)} = 1$ with $x_{\bar{j}(g_\alpha)} = 1$ and replacing $x_{j(g)} = 1$ with $x_j = 1$ in δ_0 is a feasible point of X.

Proof. Both (i) and (ii) follow from Definition 6.1 and Proposition 6.6. ■

Proposition 6.11. An α-cover inequality is non-dominated if

$$w - a_{\bar{j}(g)} \geq b \;\; \text{for each} \;\; g \in N_\alpha , \tag{6.27}$$

where w is defined as in (6.26).

Proof. Suppose that a given α-cover satisfies (6.27). We show that neither (6.24) nor (6.25) can hold true; i.e., that we cannot find $J' \subseteq J$ with α' such that

$$J' \subset J^\alpha \text{ and } \alpha' \geq \alpha \tag{6.28}$$

or $$J' = J^\alpha \bigcup \{j'\} \text{ for some } j' \in \bar{J}^\alpha \text{ and } \alpha' = \alpha + 1 . \tag{6.29}$$

First, we show that we cannot find J' that satisfies (6.28). For any $J' \subset J^\alpha$, we have, by condition (6.7), that an $x \in X$ exists such that $\left| J' \bigcap J^1(x) \right| < \alpha$, so that $\alpha' \leq \alpha - 1$. Hence, J' cannot satisfy (6.28).

We now show that we cannot find a J' that satisfies (6.29). Let $j' \in J_{g'}$. Each $j' \in \bar{J}^\alpha$ can be related to one of three disjoint sets: $g' \in \left(G_{\alpha\bar{\alpha}} \cap G_\alpha^+\right)$, $g' \in \left(G_{\alpha\bar{\alpha}} \cap G_\alpha^-\right)$, or $g' \in N_\alpha$. Each Cases 1-3, respectively, shows that we cannot obtain J' that satisfies (6.29), by including any $j' \in J_{g'}$ that is related with $G_{\alpha\bar{\alpha}} \cap G_\alpha^+$, $G_{\alpha\bar{\alpha}} \cap G_\alpha^-$, or N_α to J^α.

Case 1: Consider $g' \in \left(G_{\alpha\bar{\alpha}} \cap G_\alpha^+\right)$ and choose a $j' \in \bar{J}_{g'}^\alpha$. Let $J' = J^\alpha \bigcup \{j'\}$. Since δ_0 is in X and $\left| J^1(\delta_0) \bigcap J' \right| = \alpha$, $Z_{J'}^* = \min\left\{ \sum_{j \in J'} x_j : x \in X \right\} \leq \alpha$. Thus, in Case 1, $\alpha' \leq \alpha$.

Case 2: Consider $g' \in \left(G_{\alpha\bar{\alpha}} \cap G_\alpha^-\right)$ and choose a $j' \in \bar{J}_{g'}^\alpha$. Let $J' = J^\alpha \bigcup \{j'\}$. By Lemma 6.10, $\hat{\delta} = \delta_0 - e_{j(g_\alpha)} + e_{\bar{j}(g_\alpha)} - e_{\bar{j}(g')} + e_{j(g')}$ is in X. Since $\left| J^1(\hat{\delta}) \bigcap J' \right| = \alpha$, $Z_{J'}^* \leq \alpha$. Thus, in Case 2, $\alpha' \leq \alpha$.

Case 3. Consider $g' \in N_\alpha$ and choose a $j' \in \bar{J}^{\alpha}_{g'}$. Let $J' = J^{\alpha} \bigcup \{j'\}$. By replacing

$x_{\bar{j}(g)} = 1$ with $x_{\bar{j}(g)} = 0$ in δ_0, we obtain $\hat{\delta} = \delta_0 - e_{\bar{j}(g')}$. By condition (6.27), $\hat{\delta}$ is in X. Since $\left|J^1(\hat{\delta}) \bigcap J'\right| = \alpha$, $Z^*_{J'} \le \alpha$. Hence, in Case 3, $\alpha' \le \alpha$.

By Cases 1-3, for each $j' \in \bar{J}^{\alpha}$, $\alpha' \le \alpha$, so we cannot find a J' that satisfies (6.29). ■

Intuitively, Proposition 6.11 says that, for each GUB (i.e., $g \in N_\alpha$) that is not associated with any variable in J^α, if a feasible solution x exists in which $x_j = 0$ for all $j \in J_g$ and if the corresponding α-cover inequality is active at x, then the corresponding α-cover inequality is non-dominated.

Definition 6.2. For a given α-cover, let $g_v \in N_\alpha$ be a GUB that violates (6.27) and define $b(g_v) = b - w + a_{j(g_v)}$. Let $R(g_v) = \{ j \in J_{g_v} : a_j \ge b(g_v)\}$ be the index set of variables x_j $j \in J_{g_v}$ each of which has coefficient a_j that is greater-than-or-equal-to $b(g_v)$. An extension of the α-cover, denoted by $E(\alpha)$, is defined as $E(\alpha) = J^\alpha \bigcup R(g_v)$.

Note that $R(g_v) \ne \emptyset$; otherwise, $\text{conv}(X)$ is empty and $\text{KPG}^{\ge}$ would have no feasible solution. Also, note that, since $g_v \in N_\alpha$, $x_{j(g_v)} = 1$ in δ_0 and $j(g_v) \in R(g_v)$.

Proposition 6.12. For a given α-cover, J^α, the inequality defined as

$$\sum\nolimits_{j \in E(\alpha)} x_j \ge \alpha + 1 \tag{6.30}$$

is an $(\alpha+1)$-cover inequality. Moreover, (6.30) dominates the α-cover inequality for which it is an extension.

Proof. By way of contradiction, we first show that (6.30) is valid for $\mathrm{conv}(X)$. Suppose that $\hat{x} \in X$ exists such that $\left|E(\alpha) \cap J^1(\hat{x})\right| \le \alpha$. In order to show that this is impossible, we consider two cases: $\hat{x}_j = 0$ for all $j \in R(g_v)$ and $\hat{x}_{j(g_v)} = 1$, where g_v and $R(g_v)$ are defined as in Definition 6.2.

Case 1.1. Let $\hat{x}_j = 0$ for all $j \in R(g_v)$. Let $X^\alpha \subseteq X$ be the set of feasible points x such that $\left|J^1(x) \cap J^\alpha\right| = \alpha$. By Proposition 6.6, $\sum_{j \in J^1(x) \setminus J_{g_v}} a_j \le w - a_{j(g_v)}$ for each $x \in X^\alpha$. Since g_v violates (6.27) and $a_j < b(g_\alpha)$ for $j \in J_{g_v} \setminus R(g_v)$, $w - a_{j(g_v)} + a_j < b$ for each $j \in J_{g_v} \setminus R(g_v)$. Hence, $\sum_{j \in J^1(x) \setminus J_{g_v}} a_j + a_{j'} < b$ for each $x \in X^\alpha$ and for each $j' \in J_{g_v} \setminus R(g_v)$. Thus, there is no $\hat{x} \in X$ in Case 1.1 such that $\left|E(\alpha) \cap J^1(\hat{x})\right| \le \alpha$.

Case 1.2. Let $\hat{x}_{j(g_v)} = 1$. Suppose that $\left|E(\alpha) \cap J^1(\hat{x})\right| \le \alpha$. Since $E(\alpha) = J^\alpha \cup R(g_v)$ and $\left|R(g_v) \cap J^1(\hat{x})\right| = 1$, $\left|J^\alpha \cap J^1(\hat{x})\right| \le \alpha - 1$ must hold. However, there can be no such $\hat{x}$, since, if there were, $\min\left\{\sum_{j \in J^\alpha} x_j : x \in X\right\} \le \alpha - 1$ contradicting the feasibility of an α-cover inequality. Hence, there is no $\hat{x} \in X$ in Case 1.2 such that $\left|E(\alpha) \cap J^1(\hat{x})\right| \le \alpha$.

Together, Cases 1.1 and 1.2 show that (6.30) is valid for $\mathrm{conv}(X)$; consequently, $E(\alpha)$ satisfies condition (6.6).

In order to prove that $E(\alpha)$ is an $(\alpha+1)$-cover, we need to show that it also satisfies condition (6.7); that is, for each $\hat{j} \in E(\alpha)$, an $x \in X$ exists such that $\hat{j} \in J^1(x)$ and $\left|E(\alpha) \cap J^1(x)\right| = \alpha + 1$. $E(\alpha)$ consists of two disjoint sets: $R(g_v)$ and J^α.

Case 2.1: Let $\hat{j} \in R(g_v)$ and define $\hat{x} = \delta_0 - e_{j(g_v)} + e_{\hat{j}}$. It follows from $a_j \geq b(g_v)$ for $j \in R(g_v)$ and $b(g_v) = b - w + a_{j(g_v)}$ that $w - a_{j(g_v)} + a_j \geq b$. Hence, $\left|E(\alpha) \cap J^1(\hat{x})\right| = \alpha + 1$ for $\hat{x} \in X$.

Case 2.2: Let $\hat{j} \in J^\alpha$. By condition (6.7) and by Definition 6.2, for each $\hat{j} \in J^\alpha$, an $\hat{x} \in X$ exists such that $\left|J^\alpha \cap J^1(\hat{x})\right| = \alpha$ and $\left|R(g_v) \cap J^1(x)\right| = 1$. Thus, $\hat{x} \in X$ for which $\left|E(\alpha) \cap J^1(\hat{x})\right| = \left|J^\alpha \cap J^1(\hat{x})\right| + \left|R(g_\alpha) \cap J^1(\hat{x})\right| = \alpha + 1$.

Cases 2.1 and 2.2 show that $E(\alpha)$ satisfies (6.7). Hence, it is an $(\alpha+1)$-cover.

In order to prove that (6.30) dominates the associated α-cover inequality, we need to show that (6.30) implies it. By partitioning $E(\alpha)$ into two disjoint sets, $R(g_v)$ and J^α, (6.30) can be written as $\sum_{j \in J^\alpha} x_j \geq \alpha + \left(1 - \sum_{j \in R(g_v)} x_j\right)$. Since $\sum_{j \in R(g_v)} x_j \leq 1$, (6.30) dominates α-cover inequality (6.8). ■

Proposition 6.12 states that, for a given J^α, if a GUB g_v that violates condition (6.27) exists, we obtain an $(\alpha+1)$-cover by forming the union of $R(g_v)$ and J^α. The resulting $(\alpha+1)$-cover yields an inequality that dominates the associated α-cover inequality. Therefore, using the following procedure, we can obtain a non-dominated inequality.

Step 1. Given an α-cover, J^{α}. Find a GUB g_v that violates condition (6.27). If there is no such a GUB, terminate; the corresponding α-cover inequality is non-dominated. Otherwise, go to Step 2.

Step 2. Determine $R(g_v)$. Let $J^{\alpha} \leftarrow J^{\alpha} \cup R(g_v)$ and $\alpha \leftarrow \alpha + 1$. Go to Step 1.

Observe that for a given J^{α} and with the indices of the GUBs in G_{α} are re-numbered in non-increasing order of their $\breve{a}_{j(g)}$ values, Steps 1 and 2 together require $O(|G||J|)$ time. Consider an iteration. Defining a set of violated GUBs and determining $R(g_v)$ requires $O(|G|)$ and $O(|J|)$ time, respectively. After forming the union of $R(g_v)$ and J^{α} (i.e., $E(\alpha) = J^{\alpha} \cup R(g_v)$), we need to order the GUBs associated with $E(\alpha)$ in non-increasing order of their $\breve{a}_{j(g)}$ values. This can be done in $O(|J|)$ time using a procedure similar to Steps (8a)-(13a) of **COVER(*D*, *α*).** Thus, each iteration requires $O(|J|)$ time and there are $O(|G|)$ iterations. Note that since GUBs that violate condition (6.27) depend on the variables in J^{α}, the non-dominated inequality that we obtain depends on the sorted order of the violated GUBs considered.

Strong minimal covers for $\mathrm{KP}^{\leq}$ are non-dominated extensions of minimal covers (Balas 1975, Sherali and Lee 1995). Similarly, we define *strong* α-covers as follows:

Definition 6.3. An α-cover is a *strong* α-cover

(i) if $\alpha = \alpha^*$ or

(ii) there exists no α'-cover that strictly contains J^{α} and $|G_{\alpha'}| - |G_{\alpha}| = \alpha' - \alpha$.

6.5. Facets of conv(X)

In this section, we define the necessary and sufficient conditions for an α-cover inequality (6.8) to be a facet of $\text{conv}(X)$. First, we need to establish Lemma 6.13, which shows that G_α comprises indices of at least $(\alpha+1)$ GUBs.

Lemma 6.13. $|G_\alpha| \geq \alpha + 1$.

Proof: By condition (6.6), $|G_\alpha| \geq \alpha$. Suppose that $|G_\alpha| = \alpha$. Then, $\sum_{j \in J_g^\alpha} x_j = 1$ for each $g \in G_\alpha$ and for each $x \in X$. By Proposition 6.1, this contradicts Assumption 6.3, which requires $\text{conv}(X)$ to be full dimensional. Thus, $|G_\alpha| > \alpha$. ■

Proposition 6.14. An α-cover inequality (6.8) is a facet of $\text{conv}(X)$ if and only if

$$w - a_{j(g)} + \breve{a}_{j(g_{\alpha+1})} \geq b \qquad \text{for each } g \in G_\alpha^+ \tag{6.31}$$

$$w - a_{\bar{j}(g)} \geq b \qquad \text{for each } g \in G_{\bar{\alpha}}^- . \tag{6.32}$$

Proof: ($\Rightarrow$) We first prove the necessity of condition (6.31). Suppose that (6.8) is a facet and $w - a_{j(g)} + \breve{a}_{j(g_{\alpha+1})} < b$ for $g \in G_\alpha^+$. This implies that $\sum_{j \in J_g} x_j = 1$ for each $x \in X$ such that $\sum_{j \in J^\alpha} x_j = \alpha$. By Proposition 6.1, this means that X does not contain $|J|$ affinely independent points at which (6.8) is tight. This contradicts our assumption that (6.8) is a facet. The necessity of (6.32) can be proven using a similar argument.

($\Leftarrow$)We prove that (6.8) is a facet by identifying $|J|$ affinely independent points in X for which (6.8) is active; i.e., δ^0, δ_{gj}^1 for $g \in G_{\bar{\alpha}}^-$ and $j \in \bar{J}_g^\alpha \setminus \{\bar{j}(g)\}$, δ_g^2 for $g \in G_{\bar{\alpha}}^-$,

δ_{gj}^{3} for $g \in G_{\bar{\alpha}}^{-}$ and $j \in J_{g}^{\alpha} \setminus \{j(g_{\alpha+1})\}$, δ_{gj}^{4} for $g \in G_{\alpha}^{+}$ and $j \in \bar{J}_{g}^{\alpha}$, δ_{gj}^{5} for $g \in G_{\alpha}^{+}$, $j \in J_{g}^{\alpha} \setminus \{j(g)\}$, and δ_{g}^{6} for $g \in G_{\alpha}^{+}$. Define points as follows

(i) $\delta^{0} = \sum_{g \in G_{\alpha}^{+}} e_{j(g)} + \sum_{g \in G_{\bar{\alpha}}^{-}} e_{\bar{j}(g)}$,

(ii) $\delta_{gj}^{1} = \delta_{0} - e_{\bar{j}(g)} + e_{j}$ for each $g \in G_{\bar{\alpha}}^{-}$ and $j \in \bar{J}_{g}^{\alpha} \setminus \{\bar{j}(g)\}$,

(iii) $\delta_{g}^{2} = \delta_{0} - e_{\bar{j}(g)}$ for each $g \in G_{\bar{\alpha}}^{-}$,

(iv) $\delta_{gj}^{3} = \delta_{0} - e_{j(g_{\alpha})} + e_{\bar{j}(g_{\alpha})} - e_{\bar{j}(g)} + e_{j}$ for each $g \in G_{\alpha}^{-}$ and $j \in J_{g}^{\alpha} \setminus \{j(g_{\alpha+1})\}$,

(v) $\delta_{gj}^{4} = \delta_{0} - e_{j(g)} + e_{j} - e_{\bar{j}(g_{\alpha+1})} + e_{j(g_{\alpha+1})}$ for each $g \in G_{\alpha}^{+}$ and $j \in \bar{J}_{g}^{\alpha}$,

(vi) $\delta_{gj}^{5} = \delta_{0} - e_{j(g)} + e_{j}$ for each $g \in G_{\alpha}^{+}$ and $j \in J_{g}^{\alpha} \setminus \{j(g)\}$,

(vii) $\delta_{g}^{6} = \delta_{0} - e_{j(g)} - e_{\bar{j}(g_{\alpha+1})} + e_{j(g_{\alpha+1})}$ for each $g \in G_{\alpha}^{+}$.

Note that points (ii) and (iii) are feasible by condition (6.32); points (i), (iv) and (vi), by Lemma 6.10; and points (v) and (vii), by condition (6.31).

To complete the proof we need to show that

$$\lambda^{0}\delta^{0} + \sum_{g \in G_{\bar{\alpha}}^{-}} \sum_{j' \in \bar{J}_{g}^{\alpha} \setminus \{\bar{j}(g)\}} \lambda_{gj}^{1}\delta_{gj}^{1} + \sum_{g \in G_{\bar{\alpha}}^{-}} \lambda_{g}^{2}\delta_{g}^{2} + \sum_{g \in G_{\alpha}^{-}} \sum_{j \in J_{g}^{\alpha} \setminus \{j(g_{\alpha+1})\}} \lambda_{gj}^{3}\delta_{gj}^{3}$$
$$+ \sum_{g \in G_{\alpha}^{+}} \sum_{j \in \bar{J}_{g}^{\alpha}} \lambda_{gj}^{4}\delta_{gj}^{4} + \sum_{g \in G_{\alpha}^{+}} \sum_{j \in J_{g}^{\alpha} \setminus \{j(g)\}} \lambda_{gj}^{5}\delta_{gj}^{5} + \sum_{g \in G_{\alpha}^{+}} \lambda_{g}^{6}\delta_{g}^{6} = 0$$

and

$$\lambda^{0} + \sum_{g \in G_{\bar{\alpha}}^{-}} \sum_{j' \in \bar{J}_{g}^{\alpha} \setminus \{\bar{j}(g)\}} \lambda_{gj}^{1} + \sum_{g \in G_{\bar{\alpha}}^{-}} \lambda_{g}^{2} + \sum_{g \in G_{\alpha}^{-}} \sum_{j \in J_{g}^{\alpha} \setminus \{j(g_{\alpha+1})\}} \lambda_{gj}^{3}$$
$$+ \sum_{g \in G_{\alpha}^{+}} \sum_{j \in \bar{J}_{g}^{\alpha}} \lambda_{gj}^{4} + \sum_{g \in G_{\alpha}^{+}} \sum_{j \in J_{g}^{\alpha} \setminus \{j(g)\}} \lambda_{gj}^{5} + \sum_{g \in G_{\alpha}^{+}} \lambda_{g}^{6} = 0$$

requires that all λ^{0}, λ_{gj}^{1}, λ_{g}^{2}, λ_{gj}^{3}, λ_{gj}^{4}, λ_{gj}^{5} and λ_{g}^{6} values are zero.

Since, for each $g \in G_{\bar{\alpha}}^{-}$ and $j \in \bar{J}_{g}^{\alpha} \setminus \{\bar{j}(g)\}$, $x_{j} = 1$ in only one point (i.e., δ_{gj}^{1});

for each $g \in G_{\alpha}^{-}$ and $j \in J_{g}^{\alpha} \setminus \{j(g_{\alpha+1})\}$, $x_j = 1$ in only one point (i.e., δ_{gj}^{3}); for each $g \in G_{\alpha}^{+}$ and $j \in \bar{J}_{g}^{\alpha}$, $x_j = 1$ in only one point (i.e., δ_{gj}^{4}); and for each $g \in G_{\alpha}^{+}$ and $j \in J_{g}^{\alpha} \setminus \{j(g)\}$, $x_j = 1$ in only one point (i.e., δ_{gj}^{5}), it follows that all λ_{gj}^{1}, λ_{gj}^{3}, λ_{gj}^{4} and λ_{gj}^{5} values are 0.

Now, it is enough to consider only the remaining columns:

$$\lambda^0 \delta^0 + \sum_{g \in G_{\bar{\alpha}}^{-}} \lambda_g^2 \delta_g^2 + \sum_{g \in G_{\alpha}^{+}} \lambda_g^6 \delta_g^6 = 0 \quad \text{and} \quad \lambda^0 + \sum_{g \in G_{\bar{\alpha}}^{-}} \lambda_g^2 + \sum_{g \in G_{\alpha}^{+}} \lambda_g^6 = 0 .$$

Since, for each $g \in G_{\alpha}^{+}$, $x_{j(g)} = 0$ in only one point (i.e., δ_g^6), so that $\lambda_g^6 = 0$. In the remaining columns (corresponding to points of form (i) and (iii)), since, for each $g \in G_{\bar{\alpha}}^{-}$, $x_{\bar{j}(g)} = 0$ in only one point (i.e., δ_g^2), the corresponding $\lambda_g^2 = 0$.

Since all λ^0, λ_{gj}^1, λ_g^2, λ_{gj}^3, λ_{gj}^4, λ_{gj}^5, and λ_g^6 values are 0, $\lambda^0 = 0$. Thus, under conditions (6.31) and (6.32), an α-cover inequality is a facet of $\text{conv}(X)$. ■

Consider an α-cover inequality. Intuitively, Proposition 6.14 says that, for each GUB $g \in G$, if an $x \in X$ exists in which $x_j = 0$ for $j \in J_g$ and if the corresponding α-cover inequality is active at x, then the α-cover inequality is a facet of $\text{conv}(X)$. Furthermore, Corollary 6.15 states that, in order to establish that a given α-cover inequality is a facet, it is enough to check condition (6.31) only for the GUB $g \in G_{\alpha}^{+}$ with the largest $a_{j(g)}$ value and (6.32) only for the GUB $g \in G_{\bar{\alpha}}^{-}$ with the largest $a_{\bar{j}(g)}$ value.

Corollary 6.15. If $w - a_{j(g')} + \breve{a}_{j(g_{\alpha+1})} \geq b$ (6.33) and $$w - a_{\bar{j}(g'')} \geq b, \tag{6.34}$$

where $g' \in \arg\max\{a_{j(g)} : g \in G_{\alpha}^{+}\}$ and $g'' \in \arg\max\{a_{\bar{j}(g)} : g \in G_{\bar{\alpha}}^{-}\}$, then α-cover inequality (6.8) is a facet of $\text{conv}(X)$.

Proof. By condition (6.33), for each $g \in G_{\alpha}^{+}$ such that $a_{j(g')} \geq a_{j(g)}$,

$$b \leq w - a_{j(g')} + \breve{a}_{j(g_{\alpha+1})} \leq w - a_{j(g)} + \breve{a}_{j(g_{\alpha+1})}.$$

Since $a_{j(g)} \leq a_{j(g')}$ for each $g \in G_{\alpha}^{+}$, $w - a_{j(g')} + \breve{a}_{j(g_{\alpha+1})} \geq b$ for each $g \in G_{\alpha}^{+}$. Using a similar argument, for each $g \in G_{\bar{\alpha}}^{-}$, it can be shown that $w - a_{\bar{j}(g)} \geq b$. By Proposition 6.14, an α-cover inequality is a facet of $\text{conv}(X)$. ■

Let $V_1 \subseteq G_{\alpha}^{+}$ and $V_2 \subseteq G_{\bar{\alpha}}^{-}$ be the index subsets of GUBs that violate conditions (6.31) and (6.32), respectively. Proposition 6.17 states that, if a given α-cover inequality is not a facet of $\text{conv}(X)$, it is a facet of $\text{conv}(X(V)) \subset \text{conv}(X)$, where

$V = V_1 \bigcup V_2$,

$$\hat{V}_2 = \begin{cases} V_2 & \text{if } g_{\alpha} \notin V_1 \\ V^2 \bigcup \{g_{\alpha}\} & \text{if } g_{\alpha} \in V_1 \end{cases}$$

$$X(V) = \left\{x \in X : \sum_{j \in J_g^{\alpha}} x_j = 1 \text{ for } g \in V_1 \setminus \{g_{\alpha}\}, \sum_{j \in J_g^{\alpha}} x_j + x_{\bar{j}(g)} = 1 \text{ for } g \in \hat{V}_2\right\}.$$

In $X(V)$, each GUB $g \in V$ is replaced with an equality constraint. Moreover, $x_j = 0$ for $j \in X^0(V) = M_1 \setminus M_2$, where $M_1 = \bigcup_{g \in V} \bar{J}_g^{\alpha}$ and $M_2 = \{\bar{j}(g) : g \in \hat{V}_2\}$.

Lemma 6.16. $\dim(\text{conv}(X(\bar{V}))) = d = |J| - |X^0(V)| - |V|$.

Proof. This follows from Proposition 6.1. ■

Proposition 6.17. α-cover inequality (6.8) is a facet of $\text{conv}(X(V))$.

Proof. Not that α-cover inequality (6.8) is valid for $\text{conv}(X(V))$, because (6.8) is valid for $\text{conv}(X)$ and $\text{conv}(X(V)) \subset \text{conv}(X)$. By identifying d affinely independent points in $X(V)$ for which (6.8) is active, we prove that (6.8) is a facet of $\text{conv}(X(V))$. Consider the points defined in the proof of Proposition 6.14:

(i) δ^0; δ^1_{gj} for $g \in G^-_{\bar{\alpha}} \setminus V_2$, $j \in \bar{J}^{\alpha}_g \setminus \{\bar{j}(g)\}$;

δ^2_g for $g \in G^-_{\bar{\alpha}} \setminus V_2$; δ^5_{gj} for $g \in G^+_{\alpha}$, $j \in J^{\alpha}_g \setminus \{j(g)\}$;

(ii) δ^3_{gj} for $g \in G^-_{\alpha}$, $j \in J^{\alpha}_g \setminus \{j(g_{\alpha+1})\}$;

(iii) δ^4_{gj} for $g \in G^+_{\alpha} \setminus V_1$, $j \in \bar{J}^{\alpha}_g$; $\delta^4_{g_\alpha \bar{j}(g_\alpha)}$ if $g_\alpha \in V_1$; and δ^6_g for $g \in G^+_{\alpha} \setminus V_1$.

By Proposition 6.14, (i)-(iii) define d affinely independent points in $X(V)$. ■

Note that Proposition 6.17 is important, because we use it in Section 6.6 to show that we can obtain facets from α-cover inequalities by a lifting procedure.

6.6. Lifting procedure

We now consider a lifting procedure that lifts a given α-cover inequality (6.8) that is not already a facet. Sherali and Lee (1995) show that, in order to obtain a facet of $\text{conv}(X)$ using a lifting procedure, all of the variables associated with each GUB must be lifted simultaneously. Therefore, our lifting procedure lifts sets of variables $J_1,\ldots,J_g,\ldots,J_{|G|}$ sequentially and the variables associated with a GUB (i.e., J_g) simultaneously. We start by defining some notation related to a given α-cover, J^{α}.

For $g \in G$, let

$$\eta_g = \min\left\{\sum_{j \in J^\alpha} x_j : x \in X,\ x_j = 0 \text{ for all } j \in J_g\right\} \tag{6.35}$$

and, for $j' \in J$, let

$$\gamma_{j'} = \min\left\{\sum_{j \in J^\alpha \setminus \{j'\}} x_j : x \in X,\ x_{j'} = 1\right\}. \tag{6.36}$$

Corollary 6.18. $\gamma_{j'} = \alpha - 1$ for each $j' \in J^\alpha$; and $\gamma_{\bar{j}(g)} = \alpha$ for each $g \in \hat{V}_2$.

Proof. This follows from Lemma 6.10.

Proposition 6.19. (i) For a given α-cover, J^α,

$$\sum_{j \in J^\alpha \setminus J_{\tilde{g}}} x_j + \sum_{j \in J_{\tilde{g}}} (\eta_{\tilde{g}} - \gamma_j) x_j \geq \alpha + (\eta_{\tilde{g}} - \alpha) \qquad \tilde{g} \in G \tag{6.37}$$

is a family of valid inequalities for $\text{conv}(X)$.

(ii) Moreover, (6.37) is a facet of $\text{conv}(X(\bar{V}))$, where $\bar{V} = V \setminus \{\tilde{g}\}$.

Proof. (i) Each x_j $j \in J_{\tilde{g}}$ is either in $J_{\tilde{g}}^\alpha$ or in $\bar{J}_{\tilde{g}}^\alpha$. We prove that (6.37) is valid under three cases: $x_{j'} = 1$ for $j' \in J_{\tilde{g}}^\alpha$; $x_{j'} = 1$ for $j' \in \bar{J}_{\tilde{g}}^\alpha$; and $x_j = 0$ for all $j \in J_{\tilde{g}}$.

Case 1. For some $j' \in J_{\tilde{g}}^\alpha$, let $x_{j'} = 1$. We know by Corollary 6.18, that $\gamma_j = \alpha - 1$ for $j \in J^\alpha$. After fixing $x_{j'} = 1$ and $\gamma_{j'} = \alpha - 1$ in (6.37), we obtain

$$\sum_{j \in J^\alpha \setminus \{j'\}} x_j \geq \alpha - 1. \tag{6.38}$$

Since an α-cover inequality is valid for $\text{conv}(X)$, (6.38) is valid for $\text{conv}(X)$.

Case 2. For some $j' \in \bar{J}_{\tilde{g}}^\alpha$, let $x_{j'} = 1$. Then, (6.37) becomes

$$\sum_{j \in J^\alpha \setminus J_{\tilde{g}}} x_j \geq \gamma_{j'},$$

which is valid by (6.36).

Case 3. For all $j \in J_{\tilde{g}}$, let $x_j = 0$. Then, (6.37) becomes

$$\sum_{j \in J^{\alpha} \setminus J_{\tilde{g}}} x_j \geq \eta_{\tilde{g}},$$

which is valid by (6.35).

By Cases 1-3, (6.37) is valid for $\operatorname{conv}(X)$.

(ii) In order to prove that (6.37) is a facet of $\operatorname{conv}\left(X(\overline{V})\right)$, if $\tilde{g} \in V_1 \setminus \{g_\alpha\}$ $\left(\tilde{g} \in \hat{V}_2\right)$, we need to find $d + \left|\bar{J}_{\tilde{g}}^{\alpha}\right| + 1$ $\left(d + \left|\bar{J}_{\tilde{g}}^{\alpha}\right|\right)$ affinely independent points for which (6.37) is active. We identify affinely independent points based on whether $\tilde{g}$ violates condition (6.31) or (6.32); that is, if either $\tilde{g} \in V_1$ or $\tilde{g} \in V_2$.

Case 1. Let $\tilde{g} \in V_1$. Consider the feasible points (i)-(iii) defined in Proposition 6.17. Points (i) and (iii) satisfy (6.37) at equality, since $\alpha - 1$ variables from $J^{\alpha} \setminus J_{\tilde{g}}$ and one variable from $J_{\tilde{g}}^{\alpha}$ are fixed to 1 at each of them and $\gamma_j = \alpha - 1$ for $j \in J_{\tilde{g}}^{\alpha}$. If $\tilde{g} \neq g_\alpha$, (6.37) is active at points (ii), since $x_{j(\tilde{g})}$ and $\alpha - 1$ variables from $J^{\alpha} \setminus J_{\tilde{g}}$ are fixed to 1 at each of them and $\gamma_{j(\tilde{g})} = \alpha - 1$; otherwise, (6.37) is active at (ii), since $x_{\bar{j}(\tilde{g})}$ and α variables from $J^{\alpha} \setminus J_{\tilde{g}}$ are fixed to 1 at each of them and $\gamma_{\bar{j}(\tilde{g})} = \alpha$.

In order to prove that (6.37) is a facet of $\operatorname{conv}\left(X(\overline{V})\right)$, we define additional points under two cases:

Case 1.1. If $\tilde{g} \neq g_\alpha$, we need to define $\left|\bar{J}_{\tilde{g}}^{\alpha}\right| + 1$ more affinely independent points.

Let $\delta^7 = \delta^0 - e_{j(\tilde{g})} + \sum_{k=\alpha+1}^{\eta_{\tilde{g}}+1} (e_{j(g_k)} - e_{\bar{j}(g_k)})$

and $\delta_j^7 = \delta^0 - e_{j(\tilde{g})} + e_j + \sum_{k=\alpha+1}^{\gamma_j+1} (e_{j(g_k)} - e_{\bar{j}(g_k)})$ for each $j \in \bar{J}_{\tilde{g}}^{\alpha}$.

Cases 1.2. If $\tilde{g} = g_\alpha$, we need to define $\left|\bar{J}_{\tilde{g}}^{\alpha}\right|$ more affinely independent points; that is, δ^7 and δ_j^7 for each $j \in \bar{J}_{\tilde{g}}^{\alpha} \setminus \{j(\tilde{g})\}$.

Points δ^7 and δ_j^7 for $j \in \bar{J}_{\tilde{g}}^{\alpha}$ are feasible by (6.35) and (6.36). Point δ^7 satisfies (6.37) at equality, since $\eta_{\tilde{g}}$ variables from $J^\alpha \setminus J_{\tilde{g}}$ are fixed to 1. Each δ_j^7 satisfies (6.37) at equality, since γ_j variables from $J^\alpha \setminus J_{\tilde{g}}$ and x_j are fixed to 1. To show that these points are affinely independent, we need to show that

$$\Delta^1 + \lambda^7 \delta^7 + \sum_{j \in \bar{J}_{\tilde{g}}^{\alpha}} \lambda_j^7 \delta_j^7 = 0 \text{ and } \Delta^2 + \lambda^7 + \sum_{j \in \bar{J}_{\tilde{g}}^{\alpha}} \lambda_j^7 = 0,$$

where

$$\begin{aligned}\Delta^1 = \lambda^0 \delta^0 &+ \sum_{g \in G_{\bar{\alpha}}^- \setminus V_2} \sum_{j \in \bar{J}_g^{\alpha} \setminus \{\bar{j}(g)\}} \lambda_{gj}^1 \delta_{gj}^1 \\ &+ \sum_{g \in G_{\bar{\alpha}}^- \setminus V_2} \lambda_g^2 \delta_g^2 + \sum_{g \in G_{\alpha}^-} \sum_{j \in J_g^{\alpha} \setminus \{j(g_{\alpha+1})\}} \lambda_{gj}^3 \delta_{gj}^3 \\ &+ \sum_{g \in G_{\alpha}^+ \setminus V_1} \sum_{j \in \bar{J}_g^{\alpha}} \lambda_{gj}^4 \delta_{gj}^4 + \sum_{g \in V^1 \cap \{g_\alpha\}} \lambda_{g\bar{j}(g_\alpha)}^4 \delta_{g\bar{j}(g_\alpha)}^4 \\ &+ \sum_{g \in G_{(\alpha)}^+} \sum_{j \in J_g^{\alpha} \setminus \{j(g)\}} \lambda_{gj}^5 \delta_{gj}^5 + \sum_{g \in G_{\alpha}^+ \setminus V_1} \lambda_g^6 \delta_g^6\end{aligned}$$

and

$$\begin{aligned}\Delta^2 = \lambda^0 &+ \sum_{g \in G_{\bar{\alpha}}^- \setminus V_2} \sum_{j \in \bar{J}_g^{\alpha} \setminus \{\bar{j}(g)\}} \lambda_{gj}^1 \\ &+ \sum_{g \in G_{\bar{\alpha}}^- \setminus V_2} \lambda_g^2 + \sum_{g \in G_{\alpha}^-} \sum_{j \in J_g^{\alpha} \setminus \{j(g_{\alpha+1})\}} \lambda_{gj}^3 \\ &+ \sum_{g \in G_{\alpha}^+ \setminus V_1} \sum_{j \in \bar{J}_g^{\alpha}} \lambda_{gj}^4 + \sum_{g \in V^1 \cap \{g_\alpha\}} \lambda_{g\bar{j}(g_\alpha)}^4 \\ &+ \sum_{g \in G_{(\alpha)}^+} \sum_{j \in J_g^{\alpha} \setminus \{j(g)\}} \lambda_{gj}^5 + \sum_{g \in G_{\alpha}^+ \setminus V_1} \lambda_g^6\end{aligned}$$

implies that associated multipliers λ^0, λ_{gj}^1, λ_g^2, λ_{gj}^3, λ_{gj}^4, λ_{gj}^5, λ_g^6, λ^7, and λ_j^7 are zero.

Since for each $j \in \hat{J}$, $x_j = 1$ in only one point (i.e., δ_j^7), it follows that all $\lambda_j^7 = 0$. Since $x_j = 0$ for all $j \in J_{\tilde{g}}$ in only one point (i.e., δ^7), $\lambda^7 = 0$. According to Proposition 6.17, multipliers corresponding to remaining columns; i.e., λ^0, λ_{gj}^1, λ_g^2, λ_{gj}^4, λ_{gj}^5, and λ_g^6, are zero. Hence, if $\tilde{g} \in V_1$, (6.37) is a facet of $\mathrm{conv}\big(X(\bar{V})\big)$.

Case 2. Let $\tilde{g} \in V_2$. Consider the feasible points (i)-(iii) defined in Proposition 6.17. Inequality (6.37) is active at points (i) and δ_{gj}^3 for $g \in G_\alpha^- \setminus \{\tilde{\mathrm{g}}\}$, $j \in J_g^\alpha \setminus \{j(g_{\alpha+1})\}$, since $x_{\bar{j}(\tilde{g})}$ and α variables from $J^\alpha \setminus J_{\tilde{g}}$ are fixed to 1 at each of them and $\gamma_{\bar{j}(\tilde{g})} = \alpha$. Inequality (6.37) is active at $\delta_{\tilde{g}j}^3$ for $j \in J_{\tilde{g}}^\alpha \setminus \{j(g_{\alpha+1})\}$, since one variable from x_j $j \in J_{\tilde{g}}^\alpha \setminus \{j(g_{\alpha+1})\}$ and $\alpha - 1$ variables from $J^\alpha \setminus J_{\tilde{g}}$ are fixed to 1 at each of them and $\gamma_j = \alpha - 1$ for $j \in J_{\tilde{g}}^\alpha$. Inequality (6.37) is also active at points (iii), since $x_{\bar{j}(\tilde{g})}$ and α variables from $J^\alpha \setminus J_{\tilde{g}}$ are fixed to 1 at each of them if $\tilde{g} \neq g_{\alpha+1}$; otherwise, $x_{j(\tilde{g})}$ and $\alpha - 1$ variables from $J^\alpha \setminus J_{\tilde{g}}$ are fixed to 1 at each of them. In order to prove (6.37) is a facet, we need to define $\left|\bar{J}_{\tilde{g}}^\alpha\right|$ more affinely independent points. Let $\hat{\eta}_{\tilde{g}} = \eta_{\tilde{g}}$ if the order of $\tilde{g}$ is greater-than $\eta_{\tilde{g}}$; otherwise, $\hat{\eta}_{\tilde{g}} = \eta_{\tilde{g}} + 1$ and, for each $j \in \bar{J}_{\tilde{g}}^\alpha$. Similarly, let $\hat{\gamma}_j = \gamma_j$ if the order of $\tilde{g}$ is greater-than $\gamma_{j'}$; otherwise, $\hat{\gamma}_j = \gamma_j + 1$. Let

$$\delta^8 = \delta^0 - e_{j(\tilde{g})} + \sum\nolimits_{k=\alpha+1, g_k \neq \tilde{g}}^{\hat{\eta}_{\tilde{g}}} (e_{j(g_k)} - e_{\bar{j}(g_k)})$$

$$\delta_j^8 = \delta^0 - e_{j(\tilde{g})} + e_j + \sum\nolimits_{k=\alpha+1, g_k \neq \tilde{g}}^{\hat{\gamma}_j} (e_{j(g_k)} - e_{\bar{j}(g_k)}) \quad \text{for each } j \in \bar{J}_{\tilde{g}}^\alpha \setminus \{\bar{j}(\tilde{g})\}.$$

These points are feasible by definitions of $\eta_{\tilde{g}}$ and γ_j. Inequality (6.37) is active at δ^8, since $\eta_{\tilde{g}}$ variables from $J^\alpha \setminus J_{\tilde{g}}$ are fixed to 1. (6.37) is active at δ_j^8, since x_j and γ_j variables from $J^\alpha \setminus J_{\tilde{g}}$ are fixed to 1. Using an argument similar to that used in Case 1, it can be shown that

$$\Delta^1 + \lambda^8\delta^8 + \sum_{j\in \bar{J}_{\tilde{g}}^\alpha \setminus \{\bar{j}(\tilde{g})\}} \lambda_j^8 \delta_j^8 = 0 \text{ and } \Delta^2 + \lambda^8\delta^8 + \sum_{j\in \bar{J}_{\tilde{g}}^\alpha \setminus \{\bar{j}(\tilde{g})\}} \lambda_j^8 \delta_j^8 = 0$$

implies that associated multipliers λ^0, λ_{gj}^1, λ_g^2, λ_{gj}^3, λ_{gj}^4, λ_{gj}^5, λ_g^6, λ^8, and λ_j^8 are zero. Hence, if $\tilde{g} \in V_2$, (6.37) is a facet of $\operatorname{conv}(X(\bar{V}))$.

Since (6.37) is a facet of $\operatorname{conv}(X(\bar{V}))$ in both Cases 1 and 2 (i.e., for $\tilde{g} \in V_1$ and $\tilde{g} \in V_2$), the proof is complete. ■

Proposition 6.20. Let $V_1 \cup V_2 = \{g^1,\dots,g^v\}$ be arbitrarily ordered, where $v = |V_1| + |V_2|$. Let $V(q) = \{g^1,\dots,g^q\} \subseteq V_1 \cup V_2$, $J(q) = \bigcup_{g\in V(q)} J_g$ for $q = 1,\dots,v$, and $V(q) = J(q) = \varnothing$ for $q = 0$. For $q \in \{0,\dots,v-1\}$ suppose that

$$\sum_{J^\alpha \setminus J(q)} x_j + \sum_{j\in J(q)} \pi_j x_j \geq \pi_0$$

is valid for $\operatorname{conv}(X)$ and is a facet of $\operatorname{conv}(X(V \setminus V(q)))$. Consider step $q+1$ and calculate

$$\eta_{g^{q+1}}(q) = \min\left\{\sum_{j\in J^\alpha \setminus J(q+1)} x_j + \sum_{j\in J(q)} \pi_j x_j : x \in X,\, x_j = 0 \ j \in J_{g^{q+1}}\right\};$$

and, for each $j' \in J_{g^{q+1}}$, compute

$$\gamma_{j'}(q) = \min\left\{\sum_{j\in J^\alpha \setminus J(q+1)} x_j + \sum_{j\in J(q)} \pi_j x_j : x \in X,\, x_{j'} = 1\right\}.$$

Then, inequality

$$\sum_{J^{\alpha}\setminus J(q+1)} x_j + \sum_{j\in J(q)} \pi_j x_j + \sum_{j\in J_{g^{q+1}}} \left(\eta_{g^{q+1}}(q) - \gamma_j(q)\right) \geq \pi_0 + \left(\eta_{g^{q+1}}(q) - \pi_0\right) \qquad (6.39)$$

is valid for $\mathrm{conv}(X)$ and is a facet of $\mathrm{conv}\left(X(V \setminus V(q+1))\right)$.

Proposition 6.20 can be proven by induction from inequality (6.37) and using the argument in the proof of Proposition 6.19. Proposition 6.20 shows that if a given α-cover inequality is not a facet for $\mathrm{conv}(X)$, we can obtain a facet from it via the lifting procedure. Using notation defined in Proposition 6.20, Proposition 6.21 states that, at step q of the lifting procedure, it is enough to compute γ_j for $j \in \bar{J}^{\alpha}_{g^q}$ if $g^q \in V_1$ and γ_j for $j \in \bar{J}^{\alpha}_{g^q} \setminus \left\{\bar{j}(g^q)\right\}$ if $g^q \in \hat{V}_2$.

Proposition 6.21. Let variables associated with GUB $\bar{g}$ be the lifted at step $q+1$ of the lifting procedure. At step $q+1$ of the lifting procedure $\gamma_j(q) = \eta_{g^q}(q-1) - 1$ for $j \in J^{\alpha}_{\bar{g}}$. Moreover, $\gamma_{\bar{j}(\bar{g})}(q) = \eta_{g^q}(q-1)$ if $\bar{g} \in \hat{V}_2$.

Proof. If $q = 0$, by Corollary 6.18, $\gamma_j(q) = \alpha - 1$ for $j \in J^{\alpha}_{\bar{g}}$ and $\gamma_{\bar{j}(\bar{g})}(q) = \alpha$ if $\bar{g} \in \hat{V}_2$. Let $g^{\hat{q}}$ be the GUB lifted at each step of $\hat{q} \in Q = \{1,\ldots,q\}$. As an induction hypothesis assume that, for each $\hat{q} \in Q$,

$$\gamma_j(\hat{q}-1) = \eta_{g^{\hat{q}-1}}(\hat{q}-2) - 1 \qquad \text{for } j \in J^{\alpha}_{g^{\hat{q}}} \qquad (6.40)$$

$$\gamma_{\bar{j}(g^{\hat{q}})}(\hat{q}-1) = \eta_{g^{\hat{q}-1}}(\hat{q}-2) \qquad \text{if } g^{\hat{q}} \in \hat{V}_2. \qquad (6.41)$$

By the induction hypothesis, lifting coefficients in (6.39) are

$$\pi_j = \eta_{g^{\hat{q}}}(\hat{q}-1) - \eta_{g^{\hat{q}-1}}(\hat{q}-2) + 1 \qquad \text{for } j \in J^{\alpha}_{g^{\hat{q}}}$$

and $\qquad \pi_j = \eta_{g^{\hat{q}}}(\hat{q}-1) - \eta_{g^{\hat{q}-1}}(\hat{q}-2) \qquad$ if $g^{\hat{q}} \in \hat{V}_2$.

We re-express $\eta_{g^q}(q-1)$ in (6.39) as

$$\eta_{g^q}(q-1) = \left(\eta_{g^q}(q-1) - \eta_{g^{q-1}}(q-2)\right) + \left((\eta_{g^{q-1}}(q-2) - (\eta_{g^{q-2}}(q-3)\right) + \ldots + \left(\eta_{g^1}(0) - \alpha\right) + \alpha .$$

Thus, $\qquad \sum_{g \in Q_1} \pi_{j(g)} + \sum_{g \in Q_2} \pi_{\bar{j}(g)} = \eta_{g^q}(q-1) - \alpha + |Q_1|,$

where $\qquad G_Q = \{g^1, \ldots, g^{|Q|}\}, \quad Q_1 = G_\alpha^+ \cap G_Q, \quad$ and $\quad Q_2 = G_{\overline{\alpha}}^- \cap G_Q$.

We now show that (6.40) and (6.41) are also true for $\hat{q} = q+1$. We investigate $\overline{g}$ under three cases: $\overline{g} \in G_\alpha^+$, $\overline{g} \in G_{\overline{\alpha}}^-$, or $\overline{g} = g_\alpha$. Since (6.39) is valid for $\text{conv}(X)$, we know that $\gamma_j(q) \geq \eta_{g^q}(q-1) - 1$ for each $j \in J_{\overline{g}}^\alpha$ and $\gamma_{\bar{j}(\overline{g})}(q) \geq \eta_{g^q}(q-1)$ if $\overline{g} \in \hat{V}_2$.

Case 1: Consider $\overline{g} \in G_\alpha^+$. Let $Q_1^+ = Q_1 \cup \{\overline{g}\}$. By Lemma 6.10, for each $j \in J_{\overline{g}}^\alpha$, $\delta_1 = \delta_0 - e_{j(\overline{g})} + e_j$ is feasible with respect to X. In δ_1, $\delta_{1j} = 1$, $\delta_{1j(g)} = 1$ for $g \in G_\alpha^+ \setminus Q_1^+$, $\delta_{1j(g)} = 1$ for $g \in Q_1$, $\delta_{1\bar{j}(g)} = 1$ for $g \in Q_2$, $\delta_{1\bar{j}(g)} = 1$ for $g \in G_\alpha^- \setminus Q_2$, and the remaining variables are zero. Therefore,

$$\eta_{g^q}(q-1) - 1 \leq \gamma_j(q) \leq \sum_{g \in G_\alpha^+ \setminus Q_1^+} \delta_{1j(g)} + \sum_{g \in Q_1} \pi_{j(g)} \delta_{1j(g)} + \sum_{g \in Q_2} \pi_{\bar{j}(g)} \delta_{1\bar{j}(g)}$$

$$= \left(\alpha - |Q_1| - 1\right) + \left(\eta_{g^q}(q-1) - \alpha + |Q_1|\right) = \eta_{g^q}(q-1) - 1$$

Therefore, for each $j \in J_{\overline{g}}^\alpha$, (6.40) is true if $\overline{g} \in G_\alpha^+$.

Case 2: Consider $\overline{g} \in G_\alpha^-$. In Cases 1-2, we show that (6.40) and (6.41), respectively.

Case 2.1: Let $j \in J_{\overline{g}}^\alpha$. Define $\delta_2 = \delta_0 - e_{j(g_\alpha)} + e_{\bar{j}(g_\alpha)} - e_{\bar{j}(\overline{g})} + e_j$ in which $\delta_{2j} = 1$.

By Lemma 6.10, δ_2 is feasible for X. Using an argument similar to that used in Case 1, $\gamma_j(q) \le \eta_{g^q}(q-1)-1$ and (6.40) is true if $\bar{g} \in G_\alpha^-$.

Case 2.2: Consider $\bar{j}(\bar{g})$. In δ_0, $\delta_{0\bar{j}(\bar{g})} = 1$ and

$$\eta_{g^q}(q-1) \le \gamma_j(q) \le \sum_{g \in G_\alpha^+ \setminus Q_1} \delta_{0j(g)} + \sum_{g \in Q_1} \pi_{j(g)} \delta_{0j(g)} + \sum_{g \in Q_2} \pi_{\bar{j}(g)} \delta_{0\bar{j}(g)}$$

$$= \alpha - |Q_1| + \left(\eta_{g^q}(q-1) - \alpha + |Q_1|\right) = \eta_{g^q}(q-1).$$

Hence, (6.41) is satisfied if $\bar{g} \in G_\alpha^-$.

Case 3: Let $\bar{g} = g_\alpha$. By Case 1, (6.40) is true if $\bar{g} = g_\alpha$. We now show that (6.41) is true for $\bar{g} = g_\alpha$. Choose a $x_{\hat{j}} = 1$ such that $\hat{j} \in J_{\hat{g}}^\alpha$ and $\hat{g} \in G_\alpha^-$ and define $\delta_2 = \delta_0 - e_{j(g_\alpha)} + e_{\bar{j}(g_\alpha)} - e_{\bar{j}(\hat{g})} + e_{\hat{j}}$. Using an argument similar to previous cases, (6.41) is satisfied if $\bar{g} = g_\alpha$. ■

6.7. The separation problem

In this section, we devise a separation heuristic **SepH** to generate an α-cover inequality (6.8) to separate a fractional optimal solution to a linear relaxation of $\text{KPG}^{\ge}$, $\bar{x}$, from $\text{conv}(X)$. Note that we would like to determine an index set J^α and a value of parameter α for (6.8) that give an optimal solution to

$$\min_{1 \le \alpha \le \alpha^*} \left\{ \min_{J^\alpha \in \mathbf{J}^\alpha} \left\{ \sum_{j \in J^\alpha} \bar{x}_j - \alpha \right\} \right\}, \tag{6.42}$$

where $\mathbf{J}^\alpha$ is the set of all possible α-covers.

At each iteration **SepH** removes $\overline{H}$ from J. Then, it generates an α-cover inequality from H (i.e., $J \setminus \overline{H}$) using **Cover(**H**,**α**)** for $\alpha = \min\left\{\sum_{j \in H} x_j : x \in X\right\}$. If

$\sum_{j\in J^{\alpha}} \bar{x}_j - \alpha < 0$ holds, $\bar{x}$ violates the α-cover inequality; so that, in order to generate a violated α-cover inequality, the total sum of fractional values $\bar{x}_j$ $j \in H$ should be minimized. On the other hand, a non-trivial α-cover inequality can be generated if and only if $\sum_{g\in G_{\bar{H}}} a_{\bar{j}_H(g)} < b$ (i.e., $\alpha > 1$). This follows from the fact that even if we fix $a_{\bar{j}_H(g)} = 1$ for $g \in \overline{H}$, we should set $x_j = 1$ for some $j \in H$ in order to get a feasible solution to X. **SepH** uses theses facts along with the following KPG$^{\leq}$, which is parameterized according to the value of $\bar{b} < b$, to determine the set of variables $\overline{H}_{\bar{b}}$ that are removed from J.

$$F(\bar{b}) = \max\left\{ \sum_{j\in J} \left(\sum_{j'\in R_{\leq j}} \bar{x}_{j'} \right) z_j : \sum_{j\in J} a_j z_j < \bar{b},\ \sum_{j\in J_g} z_j \leq 1 \quad g \in G, z \in \{0,1\}^n \right\}. \quad (6.43)$$

Let $z^* \in \{0,1\}^n$ denote an optimal solution to (6.43). If $z_j^* = 1$, then $g \in G$ is chosen such that $j \in J_g$. Then, all $j' \in J_g$ such that $a_{j'} \leq a_j$ (i.e., all $j \in R_{\leq j} = \{j' \in J_g : a_{j'} \leq a_j\}$) are removed from J. Thus, $\overline{H}_{\bar{b}} = \bigcup_{j\in J^1(z^*)} R_{\leq j}$. Remember that by Lemma 6.8, if $j \notin J^{\alpha}$, then $J^{\alpha} \cap R_{\leq j} = \emptyset$. Each $R_{\leq j}$ removed from J decreases the total sum of fractional values by $\sum_{j'\in R_{\leq j}} \bar{x}_{j'}$. Hence, for a given $\bar{b}$, $\overline{H}_{\bar{b}}$ gives a subset of variables with the maximum total sum of fractional values such that $\sum_{g\in G_{\overline{H}_{\bar{b}}}} a_{j_{\overline{H}_{\bar{b}}}(g)} < b$.

SepH requires (6.43) to be solved for each $0 \leq \bar{b} < b$. Therefore, we present a dynamic program to solve (6.43) iteratively for different values of $\bar{b}$. We assume that variables associated with each GUB are listed in non-increasing order of their indices.

$$F(\bar{b}) = \max_{g\in G, k\in\{1,\ldots,|J_g|\}}\{F(g,\bar{b},k)\} \tag{6.44}$$

$$F(0,\bar{b},k) = 0 \text{ for } k = 1, 2, \ldots$$

$$F(g,\bar{b},k)=\begin{cases} F(g,\bar{b},k+1) & \text{if } a_{g(k)} \geq \bar{b} \text{ and } 1 \leq k < |J_g| \\ F(g-1,\bar{b},1) & \text{if } a_{g(k)} \geq \bar{b} \text{ and } k = |J_g| \\ \max\left\{F(g,\bar{b},k+1), F(g-1,\bar{b}-a_{g(k)},1)+\sum_{j'\in R_{g(k)\geq}} \bar{x}_{j'}\right\} & \text{if } a_{g(k)} < \bar{b} \text{ and } 1 \leq k < |J_g| \\ \max\left\{F(g-1,\bar{b},1), F(g-1,\bar{b}-a_{g(k)},1)+\sum_{j'\in R_{g(k)\geq}} \bar{x}_{j'}\right\} & \text{if } a_{g(k)} < \bar{b} \text{ and } k = |J_g| \end{cases}$$

where $F(g,\bar{b},k)$ is the objective function value in the case that the first g GUBs have not been investigated so far, the remaining knapsack size is $\bar{b}$, and the k^{th} variable associated with GUB g is investigated; and $g(k)$ is the index of the k^{th} variable associated with GUB g. In the case that $a_{g(k)} < \bar{b}$ and $1 \leq k < |J_g|$, we investigate $x_{g(k)}$: it can be either $x_{g(k)} = 0$ or $x_{g(k)} = 1$. If $x_{g(k)} = 0$, then the $(k+1)^{th}$ variable associated with GUB g is investigated; otherwise, the knapsack size is decreased by $a_{g(k)}$ and the first variable associated with GUB ($g-1$) is investigated. In the case that $a_{g(k)} < \bar{b}$ and $k = |J_g|$, similarly, it can be that either $x_{g(k)} = 0$ or $x_{g(k)} = 1$. If $x_{g(k)} = 0$, the first variable associated with GUB ($g-1$) is investigated. If $x_{g(k)} = 1$, $\bar{b}$ is decreased by $a_{g(k)}$ and the first variable associated with GUB ($g-1$) is investigated.

We now describe separation heuristic, **SepH**. It begins by solving $F(\bar{b})$ with $\bar{b} = 0$ and then increases $\bar{b}$ by 1 at each iteration, with the intent of identifying the most violated α-cover inequality.

Step 1. Initialize $\bar{b}=0$ and $H_{\alpha}=\varnothing$ for each $1\le\alpha\le\alpha^{*}$.

Step 2. Compute (6.44) to identify $H_{\bar{b}}$.

Step 3. Calculate $\alpha=\min\left\{\sum_{j\in H_{\bar{b}}} x_j : x\in X\right\}$ and set $H_{\alpha}=H_{\bar{b}}$.

If $\bar{b}<b$, increase $\bar{b}$ by 1 and go to Step 2; otherwise, go to Step 4.

Step 4. For each $1\le\alpha\le\alpha^{*}$ such that $H_{\alpha}\ne\varnothing$,

execute **Cover(**H_{α}**,** α) and calculate $\xi=\sum_{j\in J^{\alpha}}\bar{x}_j-\alpha$;

if $\xi<0$, record J^{α}.

Remark 6.2. Consider the case in which $\bar{b}=b$ and let $\hat{\alpha}=\min\left\{\sum_{j\in H_b} x_j : x\in X\right\}$. A non-trivial α-cover inequality can be obtained from a subset $J'\subseteq J$, if $\left|J'\bigcap J^{1}(x)\right|>1$ holds for each $x\in X$. Therefore, the minimum possible value that $\sum_{j\in J'}\bar{x}_j$ can take over all subsets $J'\subseteq J$ such that $\left|J'\bigcap J^{1}(x)\right|\ge 1$ is $\sum_{j\in H_b}\bar{x}_j$. This implies that for each $1\le\alpha\le\hat{\alpha}$ and for each $J^{\alpha}\in\mathbf{J}^{\alpha}$, $\sum_{j\in J^{\hat{\alpha}}}\bar{x}_j\le\sum_{j\in J^{\alpha}}\bar{x}_j$. Suppose that, for $1\le\bar{\alpha}\le\hat{\alpha}$, $\sum_{j\in J^{\alpha_b}}\bar{x}_j>\sum_{j\in J^{\bar{\alpha}}}\bar{x}_j$. By definition of H_b, this means that $\sum_{g\in\bar{G}_{\bar{\alpha}}} a_{\bar{j}(g)}\ge b$; contradicting that $\bar{\alpha}\ge 1$. Hence, $J^{\hat{\alpha}}$ and the $\hat{\alpha}$ value give an optimal solution to $\min_{1\le\alpha\le\hat{\alpha},J^{\alpha}\in\mathbf{J}^{\alpha}}\left\{\sum_{j\in J^{\alpha}}\bar{x}_j-\alpha\right\}$. ■

Proposition 6.22. SepH is of complexity $O(b\,|J|\log(|J|)+\alpha^{*}|J|^{2})$.

Proof. Step 1 requires $O(\alpha^{*})$. It is known that (6.44) can be solved in $O(|J|)$ time

(Step 2). Since Step 2 is repeated b times, Step 2 requires $O(b\,|\,J\,|)$ time. Finding α by Proposition 6.6 requires $O(|\,G\,|\log(|\,G\,|))$ time (Step 3). Since Step 2 is repeated b times, Step 3 requires $O(b\,|\,G\,|\log(|\,G\,|)+b\,|\,J\,|)$ time. By Proposition 6.9, executing **Cover(**H_α**,** α**)** requires $O(|\,J\,|^2)$ time, and calculating ξ takes $O(|\,J\,|)$ time (Step 4). Since Step 4 is repeated α^* times in the worst case, Step 4 takes $O(\alpha^*\,|\,J\,|^2)$ time. Thus, the overall time complexity of **SepH** is $O(b\,|\,J\,|+b\,|\,G\,|\log(|\,G\,|)+\alpha^*\,|\,J\,|^2)$, which reduces to $O(b\,|\,J\,|\log(|\,J\,|)+\alpha^*\,|\,J\,|^2)$. ■

6.8. Computational evaluation

In this section, we report our computational experience. We use CPLEX 11 and conduct our tests on a Dell PC (OptPlex GX620) with 3.20GZH Dual Core Processor, 2GB RAM, and 160GB hard drive.

The purpose of our tests is to evaluate the strength of inequalities (6.8), (6.30), and (6.39). The first subsection describes our test instances and the second benchmarks the strength of cuts devised in this section with that of surrogate-knapsack cuts (S-K Cuts) devised in Glover et al. (1997).

6.8.1. Test instances

The set of test instances that we use consists of ten 0-1 integer programming instances taken from MIPLIB (Table 11). We select these particular instances because they constitute a standard test bed in the field of integer programming and because they were used previously by Glover et al. (1997) to benchmark the performance of S-K cuts relative to the performance of LC cuts. Therefore, they enable us to easily benchmark our cuts with S-K and LC cuts. Columns 2-3 of Table 11 give the size of each instance in terms of the numbers of binary variables (BVs) and knapsack constraints (KPs), respectively; and columns 4-5 give the optimal objective function values of the linear programming relaxation (Z_{LP}^*) and the integer program (Z_{IP}^*), respectively. Note that most of these instances do not have the MKPG form and GUBs are not necessarily disjoint. In order to modify them to fit the MKPG form, we treat each variable that is not associated with any GUB as a member of a trivial GUB and, if a subset of GUBs is overlapping, we arbitrarily choose one to treat as a GUB and deal with others as knapsack constraints. Since the resulting test instances do not adhere to the MKPG form exactly, our cuts may not be as effective as they might be in application to the MKPG.

Table 11. Description of the test instances used in evaluating α-cover inequalities.

Instance	BVs	KPs	Z_{LP}^*	Z_{IP}^*
bm23	27	20	20.6	34
lseu	89	28	834.7	1120
mod008	319	6	290.9	307
p0033	33	16	2520.6	3089
p0201	201	134	6875.0	7615
p0282	282	242	176867.5	258411
p0291	291	253	1705.1	5223.7
p0548	548	177	315.3	8691
p2756	2756	756	2688.7	3124
sentoy	60	30	-7839.3	-7772

6.8.2. Benchmarking with S-K cuts

Glover et al. (1997) noted that the primary purpose of their computational testing was not to attempt to outperform well-established branch-and-cut codes such as CPLEX, since these codes owe their performance to a variety of enhanced techniques other than cutting planes. Rather, their goal was to determine the strength of the S-K Cuts, independent of the use of other strategies such as preprocessing. Therefore, Glover et al. (1997) benchmarks the strength of S-K cuts at the root node with that of LC cuts. Like Glover et al. (1997), we only use our strategies; our aim is to determine the relative strengths of our cuts : α-cover inequalities (6.8), non-dominated α-cover inequalities (6.30), and lifted α-cover inequalities (6.39).

We implement the following α-cover process (α-CP) to generate α-cover inequalities (6.8), non-dominated α-cover inequalities (6.30), and lifted α-cover inequalities (6.39) as needed at the root node. Each iteration of α-CP is as follows.

Step 1. Solve the linear relaxation of the overall problem to obtain a solution $\bar{x}$.

If $\bar{x}$ is integer, stop. Otherwise, go to Step 2.

Step 2. For each knapsack constraint, execute Steps 2.1 - 2.2

2.1. Invoke **SepH** to detect a violated α-cover inequality (i.e., (6.8)).

2.2. If an α-cover inequality separates $\bar{x}$ **:**

Use (6.27) to check whether it is non-dominated; and, if it is not, use the procedure defined in Section 6.4 to modify it to form a non-dominated α-cover inequality (i.e., (6.30));

Use (6.31) and (6.32) to check if the non-dominated α-cover inequality is a facet for the corresponding $\text{KPG}^{\geq}$ polytope; if it is not, modify it to be a facet (i.e., (6.39)) using Proposition 6.20.

Add the cut generated to the formulation.

Step 3. If no α **-**cover inequality is generated that separates $\bar{x}$ **,** stop.

Otherwise, return to Step 1.

Table 12 shows the number of each type of cut generated at the root node for each instance. Table 13 gives the computational results at the last iteration of α -CP and for S-K and LC cut generation. In Table 13, columns 2-4 give results obtained by using α -CP cuts; columns 5-7 give results obtained by using S-K cuts; and columns 8-10 give results obtained by using classical LC cuts. For each type of cut (i.e., α -CP, S-K, and LC), Table 13 reports three measures of performance: optimal root node solution value (Z^*_{root}); root node solution time (CPU); and the number of cuts (Cuts). Note that results for S-K and LC are obtained from Glover et al. (1997).

Table 12. Number of each cut in α -CP.

Instance	(6.8)	(6.30)	(6.39)
bm23	11	0	0
lseu	2	13	6
mod008	8	11	3
p0033	1	13	6
p0201	6	0	0
p0282	211	6	2
p0291	64	0	0
p0548	128	8	10
p2756	154	4	2
sentoy	43	0	0

Table 13 illustrates that, within a reasonable computational time, α **-**CP cuts provide stronger lower bounds than either S-K or LC cuts. In particular, α **-**CP cuts appear to yield a significant, relative advantage for solving the more challenging instances such as p0548 and p2756. However, more α-CP cuts are added in each instance than either S-K or LC cuts. Therefore, we analyze the number of α-CP cuts in more detail (Table 14).

Table 13. Benchmarking with S-K and LC cuts.

	α-CP cuts			S-K Cuts			LC Cuts		
Instance	Z^*_{root}	**CPU(sec)**	**Cuts**	Z^*_{root}	**CPU(sec)**	**Cuts**	Z^*_{root}	**CPU(sec)**	**Cuts**
bm23	22.7	0.08	11	22.7	0.1	9	22.5	0.1	1
lseu	1012.4	0.05	21	1001.2	0.3	14	999.5	0.2	13
mod008	293.3	0.06	22	291.7	0.6	5	291.3	0.2	5
p0033	2939.1	0.06	20	2902.6	0.1	15	2916.2	0.2	13
p0201	7125.0	0.03	6	7075.0	0.8	3	7075.0	0.9	2
p0282	253813.8	0.19	219	252356.0	2.5	89	180999.7	1.2	58
p0291	5055.8	0.09	64	5009.2	1.0	28	1873.8	1.3	25
p0548	7714.4	0.14	146	3883.7	8.1	158	4052.9	2.5	138
p2756	3114.3	0.69	160	2701.8	16.4	75	2701.7	10.5	68
sentoy	-7824.8	2.56	43	-7837.7	0.2	5	-7832.5	0.3	5

Table 14. Solution values and the number of cuts at different iteration of α-CP.

	Iteration 1		Iteration 2		Iteration 3	
Instance	Z_{LP}	**Cuts**	Z_{LP}	**Cuts**	Z_{LP}	**Cuts**
bm23	22.7	6	-	-	-	-
lseu	1007.6	9	1010.9	16	-	-
mod008	291.7	6	292.2	12	293.2	17
p0033	2896.0	11	2932.8	14	2939.1	19
p0201	7075.0	2	-	-	-	-
p0282	213017.9	43	249384.3	73	252494.4	95
p0291	4926.2	16	5020.5	29	5046.89	46
p0548	5005.4	58	6853.4	117	7575.7	136
p2756	2841.2	66	2953.0	113	3112.6	150
sentoy	-7834.78	3	-7828.7	25	-7826.39	31

Table 14 reports computational results after three different α-CP iterations. For each of these iterations, Table 14 gives the linear relaxation solution value (Z_{LP}) obtained at Step 1 and the cummulative number of cuts generated through that iteration. α-CP terminates before the second iteration selected on instaces bm23, lseu, and p0201, so we use "-" for the absent results. Iteration 1 gives the results at the end of the first iteration of α-CP. We choose Iterations 2 and 3 in such a way that allows us to compare the strengths of α-CP, S-K, and LC cuts. Table 14 shows that α-CP provides stronger bounds than either S-K or LC with fewer cuts. In fact, the first iteration of α-CP yields a tighter bound for each of 7 of the 10 instances than S-K cuts ultimately provide; α-CP gives tighter bounds for instances p0033 and p0291 after the second iteration and for p0282 after the third iteration. Similarly, the first iteration of α-CP yields a tighter bounds for each of 8 of the 10 instances than LC cuts ultimately provide; α-CP gives tighter bound for instances p0033 and sentoy after the second iteration. Note also that Glover et al. (1997) reports that S-K cuts are stronger than LC cuts.

CHAPTER VII

AN APPLICATION: HOUSTON SHIP CHANNEL*

Using the HSC as a test bed, Chapter III specifies 16 test instances (Table 8). Using these instances, which are SSDP instances of real size and scope, this chapter compares the efficacy of B&C, which uses α - *cover* inequalities as cuts, and B&P-D approaches. This chapter also explores the sensitivity of the system and the cost to important parameters. Part of this chapter (Section 7.3) is reprinted with permission of the IEEE from "Branch-and-Price Decomposition to Design a Surveillance System for Port and Waterway Security" by W. E. Wilhelm and E. I. Gokce.

The remainder of the chapter is organized as follows. Section 7.1 compares a branch-and-cut scheme that uses inequalities (6.8), (6.30), and (6.39) as cuts with branch-and-cut settings of CPLEX 11 that use either classical lifted cover (LC) cuts or GUB cover cuts. Section 7.2 compares the cuts (i.e., (6.8), (6.30), and (6.39)) with B&P-D. Section 7.3 presents the suggested surveillance system design for HSC. Finally, Section 7.4 conducts a sensitivity analysis.

7.1. Using a B&C approach to solve SSDP

We tested three different cut-generation strategies using instances described in Table 8. The first strategy (S1) involves detecting a violated α -cover inequality for each

$\text{KPG}^{\geq}$ substructure, and adding each of them without modification (i.e., without invoking the non-domination check or lifting). The second strategy (S2) detects a violated α-cover inequality for each $\text{KPG}^{\geq}$ substructure and adds it after modifying it by lifting to be a facial inequality. The third strategy (S3) is the same as S2, except it adds only the most violated α-cover inequality after lifting it (if necessary) to be a facial inequality. If no violated inequality is found, we branch on the most fractional variable. We also apply the best-bound node-selection strategy. Table 15 gives results for three runs using S1 (columns 2-4), S2 (columns 5-7), and S3 (columns 8-10). For each strategy (i.e., S1, S2, and S3), Table 15 repeats three measures of performance: the number of B&B-nodes searched (Node); the total number of cuts added (Cuts); and the run time required to find the optimal integer solution (CPU).

Table 15. Computational results for different cut generating strategies**.**

	S1			**S2**			**S3**		
N	**Node**	**Cuts**	**CPU(sec)**	**Node**	**Cuts**	**CPU(sec)**	**Node**	**Cuts**	**CPU(sec)**
1	11	57	0.03	11	50	0.03	13	54	0.08
2	45	225	0.13	29	119	0.08	21	78	0.16
3	11	55	0.05	5	48	0.03	5	31	0.06
4	33	181	0.20	15	98	0.08	15	64	0.13
5	1	97	0.05	1	91	0.03	1	28	0.08
6	3	152	0.06	3	147	0.05	5	49	0.16
7	5	98	0.06	5	98	0.06	5	23	0.09
8	13	242	0.16	5	166	0.06	3	53	0.20
9	37	109	0.08	25	105	0.05	25	51	0.12
10	21	105	0.06	21	103	0.05	21	58	0.12
11	31	125	0.09	21	123	0.08	19	76	0.23
12	11	139	0.08	11	131	0.06	15	86	0.20
13	47	458	0.24	47	458	0.24	37	71	0.25
14	35	271	0.19	31	244	0.17	61	84	0.39
15	673	1147	3.84	425	731	3.47	176	468	2.23
16	17	273	0.17	17	255	0.14	31	155	0.64

The run times for S1, S2, and S3 are negligible on all instances. Strategy S1 requires 20% more B&B-nodes and adds 15% more cuts than those required by S2. This illustrates that generating stronger inequalities (facets) helps to close the gap between Z^*_{LP} and Z^*_{IP}, especially for the instances N = 2, 4, 8, and 15. Also, note that on all instances, S2 takes less run time S1, so that lifting α-cover inequalities reduces run time. Strategy S3 requires about the same number of B&B-nodes as S2 (except for instance 15 in which it requires considerable fewer nodes), but 50% fewer cuts, showing the strength of the lifted α-cover inequalities (6.39).

To benchmark S1, S2, and S3, we compare them with the B&B routine of CPLEX using only LC cuts, using only GUB covers, and using no cuts at all. In all runs, we have turned off the CPLEX pre-processing capability, so that the CPLEX results would be comparable with those of our procedures. Table 16 gives results. In Table 16, columns 2-4 give CPLEX results obtained by using LCs; columns 5-7 give results obtained by using GUB covers; and columns 8-9 give results obtained using CPLEX with no cuts. For LCs and GUB covers, Table 16 repeats three measures of performance: the number of B&B-nodes searched (Nodes); the number of cuts added (cuts); and the run time required to find the optimal integer solution (CPU). For CPLEX B&B, Table 16 reports the number of B&B-nodes needed to reach optimality (or, the number of B&B-nodes searched within 60,000 seconds, our time limit) and the run time required to find the optimal integer solution.

Strategies S1 and S2 require less run time than LCs, except for one instance (N = 15) and S3 requires approximately the same run time as LCs. LCs, S1, S2, and S3 are all faster than GUB covers. On all instances, each of S1, S2, and S3 requires considerably fewer nodes than either LCs or GUB covers. Both S1 and S2 generate more cuts than either LCs or GUB covers. This is expected, since both S1 and S2 add a violated α-cover inequality for each $KPG^{\geq}$ substructure. On the other hand, S3 requires considerably fewer cuts than LCs, except for two instances (N = 2, 15). GUB covers add fewer cuts, but they are not stronger than the cuts generated by S3, since the number of B&B nodes searched is considerable more than for S3. Rather, the most likely reason for this is that CPLEX cannot find violated GUB cuts to tighten successfully. Note that 99% of the GUB covers are added at the root node and only a few are added after branching.

Table 16. CPLEX results for HSC instances – LC and GUB covers.

	B&B-LC			B&B – GUB Covers			B&B		B&P-D	
N	Node	Cuts	CPU(sec)	Node	Cuts	CPU(sec)	Node	CPU(sec)	Node	CPU(sec)
1	165	54	0.19	296	36	0.30	106,630	12.92	1	2.50
2	111	71	0.16	151	50	0.16	2,135,618	327.92	1	8.78
3	210	68	0.25	1141	40	0.95	232,269	36.42	1	14.17
4	650	140	0.70	1087	57	0.94	5,984,1137	14,039.23	5	2893.99
5	1	88	0.03	155	93	0.17	441,217	90.61	1	13.19
6	70	118	0.11	105	105	0.17	35,050,870	8,858.55	3	1882.54
7	15	72	0.05	128	88	0.16	1,880,246	523.00	1	64.16
8	45	98	0.11	158	128	0.22	137,266,419	60,000.02	1	2494.44
9	17	83	0.06	347	76	0.31	14,294,251	2,554.27	1	1.91
10	55	82	0.09	131	81	0.14	239,891,803	60,000.17	1	3.69
11	112	96	0.17	1657	96	1.61	205,864,226	60,000.33	3	13.97
12	615	145	0.80	4017	94	3.59	177,343,880	60,000.09	5	278.21
13	130	172	020	258	151	0.68	133,389,729	20,571.05	7	230.05
14	85	182	0.27	113	182	0.27	132,713,241	60,000.02	3	373.53
15	344	152	0.55	6930	171	7.72	135,053,281	60,000.00	3	2074.62
16	268	243	0.47	624	179	0.77	108,500,672	60,000.00	3	5151.07

7.2. Comparison with B&P-D

This section compares B&P-D with our B&C, which uses the inequalities derived in Chapter VI. Columns 10-11 of Table 16 give the number of B&B nodes and the run time, respectively, required by B&P-D to find an integer optimal solution.

On all instances, B&P-D requires significantly less number of nodes than B&C. B&P-D is able to solve 8 of these 16 instances at the root node and all of them within 7 B&B nodes. It is able to solve these 16 instances faster than CPLEX B&B. However, when the number of clones increases, run time that B&P-D spent to solve RMP increases, putting it at a disadvantage. Both LCs and GUB covers improve on its run times. However, new strategy S2 is the fastest of the methods.

7.3. Surveillance system design for the HSC

This section presents the surveillance system design that our model suggests for the HSC. In Section 3.2.4, we identify surveillance points under two different assumptions. Figure 12 displays the design for the first assumption, which assumes that any sensor that is capable of observing the point would also be able to observe the entire line and its vicinity. Figure 13 displays the design for the second assumption, which requires that each surveillance point be observed by sensor(s) located on the same side of the channel.

7.4. Sensitivity analysis

In this section we evaluate the robustness of the optimal surveillance system

Figure 12. Optimal surveillance system designs for instance 8.

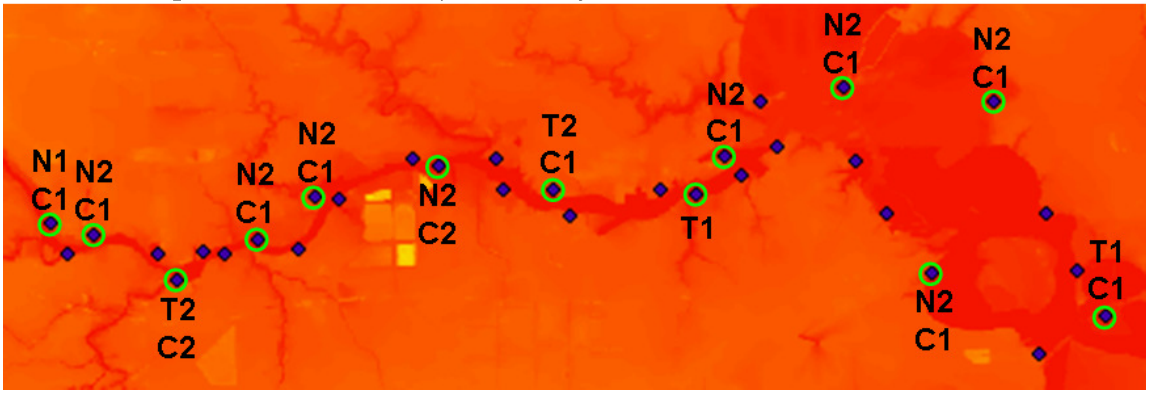
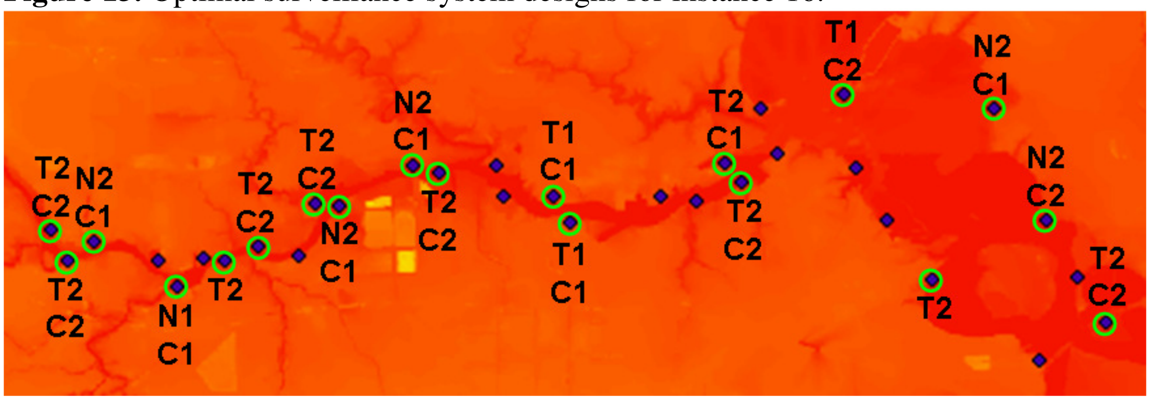


Figure 13. Optimal surveillance system designs for instance 16.



design to detection probability requirements (1-t_{es}), maintenance cost, and land cost. In Section 3.2.6 (1-t_{es}) values require detection probability of at least 0.95 (0.965 on average) at each surveillance point. We change the detection probability requirement at each surveillance point by -2.0%, -1.5%, -1%, -0.5%, 0.5%, 1.0%, 1.5%, and 2.0%. Figures 14 and 15 display the results of this analysis for instances 8 and 16, respectively. In Figure 14 (Figure 15) columns 1-21 (1-32) represent sensor locations; columns 22 and 23 (33 and 34) give the optimal number of locations and sensors prescribed. Each row in Figures 14 and 15 denotes the solution (i.e., sensor combinations) prescribed for the associated % change of detection probability requirement (1-t_{es}). Note that Figure 15 stops at a 0.8% increase of (1-t_{es}), because instances associated with larger increases (i.e., as (1-t_{es}) approaches 1.0) are infeasible. The implication is that, in practice, it becomes very costly to require (1-t_{es}) values that are close to 1.0.

Figures 14-15 show that the optimal system design is relatively insensitive to changes in (1-t_{es}) until its value approaches to 1.0. If (1- t_{es}) is less than 0.99, changing (1- t_{es}) values by 0.5% requires modifying sensor combinations at three or four sensor locations (i.e., 9% of sensor locations) on average. However, in order to increase (1- t_{es}) to a value close to 1.0, the system must use almost all sensor locations and upgrade sensor combinations at many locations.

Figure 14. Sensitivity analysis for instance 8.

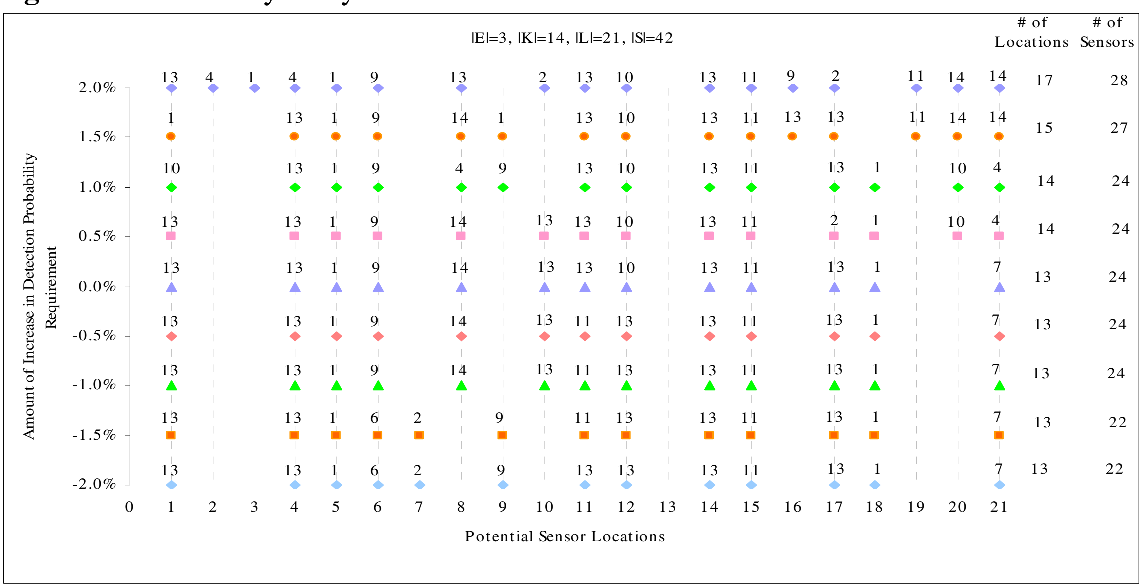


Figure 16 displays the percentage change in the optimal cost value for each change of (1-t_{es}) for instances 8 and 16. For both instances, optimal cost values increase with the detection probability requirement, significantly as (1-t_{es}) approaches 1.

Figure 15. Sensitivity analysis for instance 16.

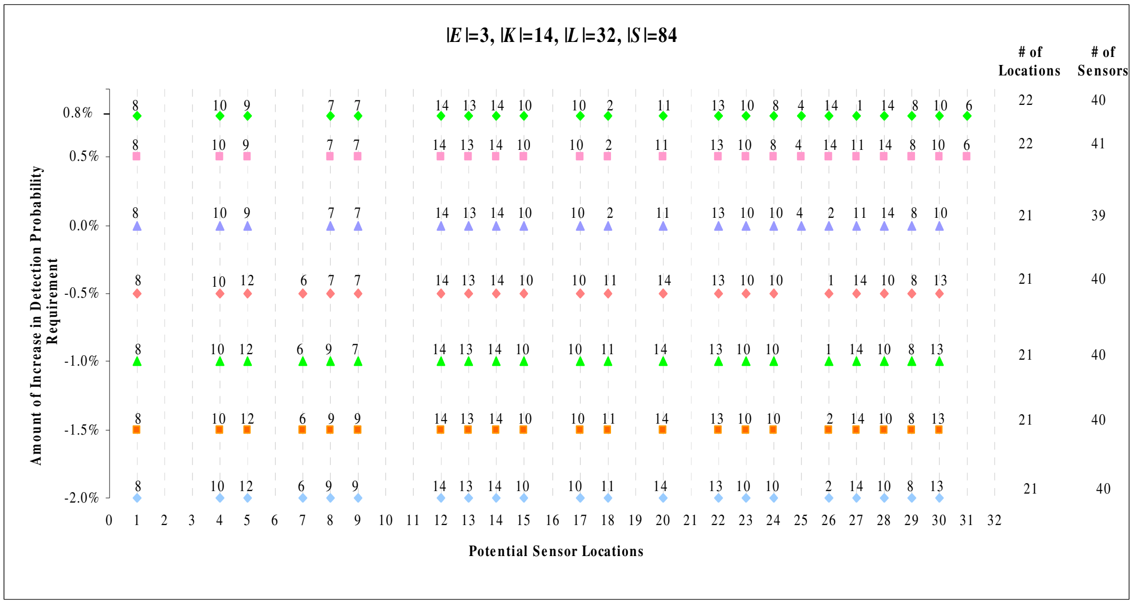
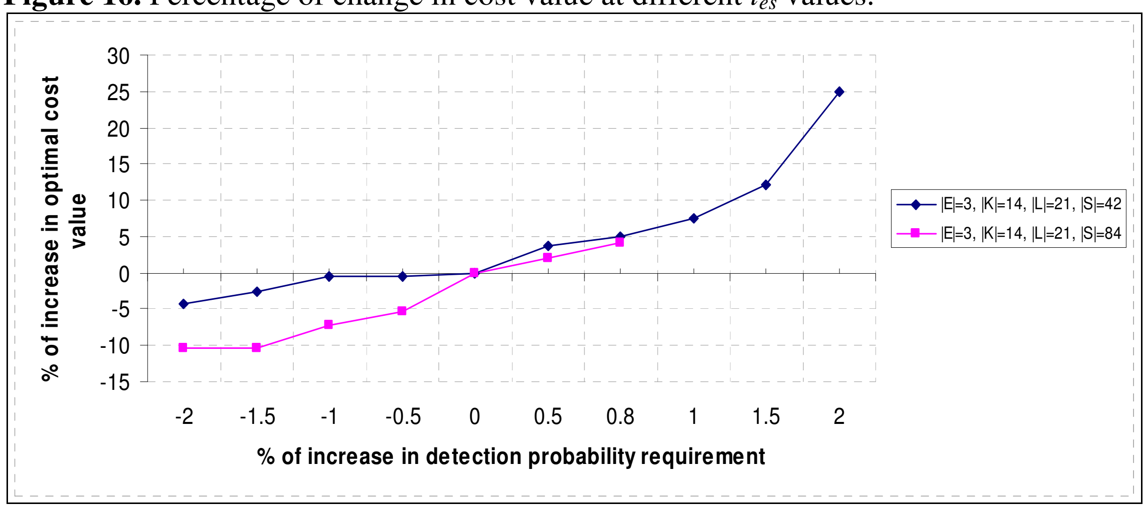


Figure 16. Percentage of change in cost value at different t_{es} values.



To evaluate the effect of annual maintenance cost on the optimal surveillance system design, we compare B&P-D-prescribed designs for annual maintenance costs that are 1%, 5%, 10%, and 20% of purchasing and installing costs. The optimal design is the same for instances 8 and 16 under these four conditions, indicating that designs are not sensitive to maintenance cost.

To evaluate the effect of land cost on the optimal surveillance system design, we change land cost by -10%, -5%, 0%, 5%, 10%, 15%, and 20%. Again, the optimal design does not change for instances 8 and 16 under these seven conditions.

CHAPTER VIII

CONCLUSIONS AND FUTURE RESEARCH

This dissertation synthesizes a methodology to prescribe a surveillance system design (SSD) to provide the required level of surveillance for ports and waterways. It achieves its purpose in three related parts: formulation of the SSD problem (SSDP) for ports and waterways, branch-and-prince decomposition (B&P-D) and branch-and-cut (B&C) solution methodologies to solve large-scale SSDPs.

8.1. Conclusion and future research on SSDP formulation

In the first part of this dissertation, we formulate a linear integer programming model to prescribe a minimum cost surveillance system design for port and waterway security. Our model represents relevant practical considerations, including the irregular shapes of ports and waterways (e.g., long, narrow, and meandering paths); the line-of-sight requirement between a sensor and a surveillance point; and the capabilities of each sensor type, which depend upon time of day, weather conditions, and distance to a surveillance point. The form of this model is a multidimensional knapsack problem with generalized upper bound constraints (i.e., MKGP). Surveillance system obtained by solving this model generally requires multiple sensors to observe each surveillance point. In the operation of multiple sensors, we may encounter inconsistent sensor observations. Our future research will contribute by proposing a decision scheme to determine the right interpretations of sensor outputs when conflict arises. Another question is the fault tolerance capability (FTC) of a surveillance system to measure its robustness to sensor failures and a methodology to determine the number of tolerable faults. Our future research includes defining FTC and modifying the current model to consider the possibility of sensor failures. It is important to note that sensitivity analysis shows that cost is relatively insensitive to changes in detection probability (unless the requirement approaches to 1.0), maintenance cost, and land cost. In addition, depending upon the elevations and terrain features in other application areas, it may be of interest to study tower height as an additional experimental factor. Moreover, the proposed approach could be adapted/refined for related applications such as border patrol and underwater surveillance.

8.2. Conclusion and future research on B&P-D

The second part of this dissertation proposes a B&P-D solution procedure to solve the SSDP. We first present three B&P-Ds and study the theoretical relationships among the bounds that these formulations provide with the goal of identifying a B&P-D formulation that provides strong bounds for SSDP. These B&P-Ds have subproblems (SPs) that can be solved in pseudo-polynomial time. We compare the bounds that can be obtained from B&P-Ds and Lagrangian methods (i.e., Lagrangian relaxation (LR), Lagrangian decomposition (LD)). B&P-D provides the same bound as LD, which is well known to provide tighter bounds than LR. However, Lagrangian approaches generally use procedures based on subgradient optimization to search for the optimal Lagrange multipliers. Since these approaches may not find the optimal multipliers - if they exist - and usually stop with a "near optimal" solution, Lagrangian methods are not guaranteed to prescribe optimal solutions. B&P-D overcomes the possibility that optimal Lagrange multipliers may not be found, guaranteeing the best bound possible. Finally, we consider improving the lower bound by incorporating a surrogate constraint in master problem (MP). Our results show that incorporating a surrogate constraint in the corresponding MP does not tighten B&P-D bounds.

With the goal of identifying an effective means of implementing B&P-D, we computationally evaluate 72 cases, each of which is a combination of a decomposition and an implementation technique. Computational tests provide considerable insight into the influence that each factor (B&P-D formulation, cost assignment, restricted MP (RMP) formulation and surrogate constraint) has on run time. Our results show that subproblem types (i.e., knapsack problem (KP) or multiple-choice knapsack problem (MCKP)), cost assignment and RMP formulation have significant affect on run time. However, including either generalized upper bound constraints (GUBs) or a surrogate constraint in RMP has no affect on run time. Based on our analysis we define the default B&P-D implementation technique for solving the surveillance system design problem as follows:

Level 3 of Factor 1: B&P-D_3 = no GUBs in RMP + MCKP.

Level 1 of Factor 2: uniform cost assignment with equality constraints.

Level 3 of Factor 3: using equality constraint (4.22) only for clones with $a_{ij} > 0$;

aggregated equality constraint (4.23) for clones with $a_{ij} = 0$.

Level 1 of Factor 4: RMP without any surrogate constraint.

Furthermore, we describe three branching rules (branching on the most fractional variable (B1); GUBs (B2); and special ordered set (B3)) and two heuristics (construction heuristic (CH), and construction and improvement heuristic (CIH)) for generating an initial basic feasible solution at each node of B&B tree. Using default B&P-D formulation, we test alternative combinations of these branching rules and heuristics. Our results show that CIH heuristic in combination with branching rule B3 generally requires less run time than alternatives, and we define these implementation techniques in our default B&P-D.

Computational tests fulfill our third objective by showing that the default B&P-D requires significantly less run time than CPLEX branch-and bound (B&B) and providing considerable insight into the influence that each parameter (i.e., experimental factor) has on run time. Tests also show that B&P-D provides very strong bounds; but significant amount of run time is spent for solving RMP. Motivated by these results, our future research on B&P-D will contribute by incorporating cutting planes to tighten RMPs, making them less challenging to solve. Also, stabilization methods could be adapted in order to improve the convergence of the proposed B&P-D approach.

8.3. Conclusion and future research on B&C

The third part of this dissertation proposes a B&C procedure to solve the SSDP. We first devise a set of valid inequalities, called α-cover inequalities, for $\text{conv}(X)$ along with a polynomial-time procedure to generate such an inequality. Then, we establish non-dominance relationships between α-cover inequalities and discuss a procedure to obtain a non-dominated α-cover inequality. Later, we define the necessary and sufficient conditions for a non dominated α-cover inequality to define a facet of $\text{conv}(X)$. We develop a lifting procedure (6.39). It lifts variables $J_1,...,J_{|G|}$ sequentially and the variables associated with a GUB (i.e., $J_g \quad g \in G$) simultaneously. Furthermore, we show that, if an α-cover inequality is not a facet of $\text{conv}(X)$, we can obtain a facet from it via (6.39). Finally, we present a separation heuristic **SepH** to generate a violated α-cover inequality to cut off a fractional solution to the linear relaxation of knapsack problem with GUBS ($\text{KPG}^{\geq}$). Computational tests shows that cuts generated by α-cover procedure (α-CP) (i.e., α-cover, non-dominated α-cover, and lifted α-cover inequalities) provide tighter cuts than either surrogate knapsack (S-K) or lifted cover (LC) cuts and using α-CP to generate cuts for multidimensional $\text{KPG}^{\geq}$ ($\text{MKPG}^{\geq}$) solves our integer test instances in less run time. Tests also show that strong inequalities (i.e., facets) serve well to close the gap between Z^*_{LP} and Z^*_{IP}. Future research could contribute, for example, by devising a sequence-independent lifting procedure for α-cover inequalities or generalizing α-cover inequalities directly for the convex hull of the integer solutions that are feasible with respect to all knapsacks in $\text{MKPG}^{\geq}$. Our research continues along these lines.

We also compare B&P-D with our B&C, which uses the inequalities derived in Chapter VI. Our results show that B&C strategy S2, which detects a violated α-cover inequality for each $\text{KPG}^{\geq}$ substructure and adds it after modifying it by lifting to be a facial inequality, is the fastest of the methods.

REFERENCES

Akbar, M. D. M., M. S. Rahman, M. Keykobad, E. G. Manning, G. C. Shoja. 2006. Solving the multidimensional multiple-choice knapsack problem by constructing convex hulls. *Computers and Operations Research* **33** 1259-1273.

Akyildiz, I. F., W. Su, Y. Sankarsubramaniam, E. Cayirci. 2002. Wireless sensor networks: A survey. *Computer Networks* **38** 393-422.

Alonso, A. A., C. E. Frouzakis, I. Kevrekidis. 2004. Optimal sensor placement for state reconstruction of distributed process systems. *AICHE Journal* **50**(7) 1438-1452.

Armstrong, R. D., D. S. Kung, P. Sinha, A. A. Zolterns. 1983. A computational study of multiple-choice knapsack algorithm. *ACM Transactions on Mathematical Software* **9** 184-198.

Bagajewicz, M. J., E. Cabrere. 2002. New MILP formulation for instrumentation network design and upgrade. *AICHE Journal* **48**(10) 2271-2282.

Balas, E. 1975. Facets of the knapsack polytope. *Mathematical Programming* **8** 146-164.

Balas, E., E. Zemel. 1978. Facets of the knapsack polytope from minimal covers. *SIAM Journal of Applied Mathematics* **34** 119-148.

Bazaraa, M. S., J. J. Jarvis, H. D. Sherali. 1990. *Linear Programming and Network Flows*. Wiley-Interscience, New York.

Ben-Zvi, T., J. Nickerson. 2007. Sensor placement learning automata. *Proceedings of INFORMS International Conference*, Puerto Rico, 67.

Berry, J., W. E. Hart, C. A. Phillips, J. Uber. 2004. A general integer-programming-based framework for sensor placement in municipal water networks. *Proceedings of Sixth Annual Symposium on Water Distribution Systems Analysis*, Salt Lake City, Utah, 1-10.

Bertsimas, D., R. Demir. 2002. An approximate dynamic programming approach to multidimensional knapsack problems. *Management Science* **48**(4) 550-65.

Bottino, A., A. Laurentini. 2004. Optimal positioning of sensors in 2D. *Proceedings of CIARP 2004*, Pueblo, Mexico, 53-58.

Bottino, A., A. Laurentini. 2005. Optimal positioning of sensors in 3D. *Lecture Notes in Computer Science* **3773** 804-812.

Cavalier, T., W. Conner, E. del Castillo, S. Brown. 2007. Heuristic algorithm for minimax sensor location in the plane. *European Journal of Operational Research* **183**(1) 42-55.

Chakrabarty, K., S. S. Iyengar, H. Qi, E. Cho. 2002. Grid coverage for surveillance and target location in distributed sensor networks. *IEEE Transactions on Computers* **51**(12) 1448-1453.

Chardaire, P., G. P. McKeown, J. A. Maki. 2001. Application of GRASP to the multiconstraint knapsack problem**.** *Lecture Notes in Computer Science* **2037** 30-39.

Chu, P. C., J. E. Beasley. 1998. A genetic algorithm for the multidimensional knapsack problem. *Journal of Heuristics* **4** 63-86.

Clark, D. 2004. Low-cost device for monitoring is set for market. *Wall Street Journal*. (September 20) B5.

Closed Circuit Television Camera. 2008. Retrieved September 25, 2008, http://en.wikipedia.org/ wiki/Closed-circuit_television.

Congressional Research Service Report. 2008. Terrorist attack on USS Cole: Background and issues for congress. Retrieved September 26, 2008, http://www.fas.org/irp/crs/RL32058.pdf.

Cormen, T. H., C. E. Leiserson, R. L. Rivest. 1990. *Introduction to Algorithms*. MIT Press and McGraw-Hill, Boston.

Crama, Y., J. Mazzola. 1994. On the strength of relaxations of multidimensional knapsack problems. *INFOR* **32** 219-225.

Crowder, H. P., E. L. Johnson, M. W. Padberg. 1983. Solving large scale zero-one linear programming problems. *Operations Research* **31** 803-834.

Dantzig, G. 1957. Discrete-variable extremum problems. *Operations Research* **5** 266-277.

Dudzinski, K., S. Walukiewicz. 1987. Exact methods for the knapsack problem and its generalization. *European Journal of Operational Research* **28** 3-21.

Esri. 2008. Retrieved September 25, 2008, http://www.esri.com.

Feo, T. A., M. G. C. Resende. 1995. Greedy randomized adaptive search procedures. *Journal of Global Optimization* **6**(2) 109-133.

Fernandes, A. M. , A. B. Utkin, A. V. Lavrov. 2006. Optimisation of location and number of lidar apparatuses for early forest fire detection in hilly terrain. *Fire Safety Journal* **41**(2) 144-154.

Frangioni, A. 2005. About Lagrangian methods in integer optimization. *Annals of Operations Research* **139**(1) 163-193.

Fréville, A. 2004. The multidimensional 0–1 knapsack problem: An overview. *European Journal of Operational Research* **155** 1-21.

Fréville, A, S. Hanafi. 2005. The multidimensional 0-1 knapsack problem – bounds and computational aspects. *Annals of Operations Research* **139**(1) 195-227.

Fréville, A., G. Plateau. 1994. An efficient preprocessing procedure for the multi-dimensional 0-1 knapsack problem. *Discrete Applied Mathematics* **49** 189-212.

Gabrel, V., M. Minoux. 2002. A scheme for exact separation of extended cover inequalities and application to multidimensional knapsack problems. *Operations Research Letters* **30** 252-264.

Garey, M. R., D. S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-completeness*. Freeman, San Francisco, CA.

Gavish, B., H. Pirkul. 1985. Efficient algorithms for solving multi-constraint zero-one knapsack problems to optimality. *Mathematical Programming* **31**(1) 78-105.

Geoffrion, A. M. 1974. The Lagrangian relaxation and its uses in integer programming. *Mathematical Programming* **2** 82-114.

Gilmore, P. C., R. E. Gomory. 1966. The theory and computation of knapsack function. *Operations Research* **14** 1045-1074.

Glover, F. 1968. Surrogate constraints. *Operations Research* **16**(4) 741-749.

Glover, F. 1975. Surrogate constraint duality in mathematical programming. *Operations Research* **23**(3) 434-451.

Glover, F., H. D. Sherali. 2008. Second-order cover inequalities. *Mathematical Programmin*g Series A **114**(2) 207-234.

Glover, F., H. D. Sherali, Y. Lee. 1997. Generating cuts from surrogate constraint analysis for zero-one and multiple choice programming. *Computer Optimization and Applications* **8** 151-172.

Google Earth. 2008. Retrieved September 25, 2008, http://earth.google.com.

Greenberg, H. J., W. P. Pierskalla. 1970. Surrogate mathematical programming. *Operations Research* **18**(5) 924-939.

Gu, Z., G. L. Nemhauser, M. W. P. Savelsbergh. 1998. Cover inequalities for 0-1 linear programs: Complexity. *Informs Journal on Computing* **10** 427-437.

Gu, Z., G. L. Nemhauser, M. W. P. Savelsbergh. 1999. Cover inequalities for 0-1 linear programs: Computation. *Informs Journal on Computing* **11** 117-123.

Guignard, M., S. Kim. 1987a. Lagrangean decomposition: A model yielding stronger lagrangean bounds. *Mathematical Programming* **39** 215-228.

Guignard, M., S. Kim. 1987b. Lagrangean decomposition for integer programming: Theory and applications. *R.A.I.R.O. Recherche Perationelle/Operations Research* **21**(4) 307-323.

Guignard, M., G. Plateau, G. Yu. 1989. An application of Lagrangean decomposition to the 0–1 biknapsack problem. Working paper, Wharton School, Vancouver, Canada.

Hammer, P. L., E. L. Johnson, U. N. Peled. 1975. Facets of regular 0-1 polytopes. *Mathematical Programming* **8** 179-206.

Hanafi, S., A. Fréville. 1998. An efficient tabu search approach for the 0-1 multi-dimensional knapsack problem. *European Journal of Operational Research* **106** 659-675.

Hanafi, S., A. Fréville, A. El Abedellaoui. 1996. Comparison of heuristics for the 0-1 multidimensional knapsack problem. I.H. Osman, J.P. Kelly, ed. *Meta-Heuristics: Theory and Applications*. Kluwer, Boston, MA, 449-465.

Healy, W. C. 1964. Multiple choice programming. *Operations Research* **12** 122-138.

Hifi, M., M. Michrafy, A. Sbihi. 2004. Heuristic algorithms for the multiple-choice multidimensional knapsack problem. *Journal of Operational Research Society* **55**(12) 1323-1332.

Hifi, M., M. Michrafy, A. Sbihi. 2006. A reactive local search-based algorithm for the multiple-choice multidimensional knapsack problem. *Computational Optimization and Applications* **33** 271-285.

Johnson, E. L., M. M. Kostreva, U. H. Suhl. 1985. Solving 0-1 integer programming problems arising from large scale planning models. *Operations Research* **33**(4) 803-819.

Johnson, E. L., M. D. Padberg. 1981. A note on the knapsack problem with special ordered sets, *Operation Research Letters* **1**(1) 18-22.

Karwan, M. H., R. Rardin. 1979. Some relationships between Lagrangian and surrogate duality in integer programming. *Mathematical Programming* **1** 320-334.

Kellerer, H., U. Pferschy, D. Pisinger. 2004. *Knapsack Problems*. Springer, Berlin.

Khan, S., K. F. Li, E. G. Manning, M. D. M. Akbar. 2002. Solving the knapsack for adaptive multi-media systems. *Studia Informatica* **2**(1) 154-174.

Kharchenko, V., V. Vasylyev. 2002. Application of the intellectual decision making system for vessel traffic control. *Proceedings of the 14th International Conference on Microwaves, Radar and Wireless Communications*, **2** 639-642.

Kim, H. S., S. Park. 2006. An Optimal Algorithm for the Sensor Location Problem to Cover Sensor Networks. *KIIE/KORMS 2006 Spring Combined Conference Proceedings*, KAIST, Daejeon, South Korea, 1-8.

Li, V. C. 2005. Tight oscillations tabu search for multidimensional knapsack problems with generalized upper bound constraints. *Computers and Operations Research* **32**(11) 2843-2852.

Li, V. C., G. L. Curry. 2005. Solving multidimensional knapsack problems with generalized upper bound constraints using critical event tabu search. *Computers and Operations Research* **32**(4) 825-848.

Li, V. C., G. L. Curry, E. A. Boyd. 2004. Towards the real time solution of strike force asset allocation problems. *Computers and Operations Research* **31**(2) 273-291.

Lin, E. Y. H. 1998. A bibliographical survey on some well-known non-standard knapsack problems. *INFOR* **36**(4) 274-317.

Lin, F. Y. S., P. L. Chiu. 2005. A near-optimal sensor placement algorithm to achieve complete coverage/discrimination in sensor networks. *IEEE Communications Letters* **9**(1) 43-45.

Lorie, J., L. J. Savage. 1955. Three problems in capital rationing. *Journal of Business* (October).

Lougee-Heimer, R. 2001. A note on coefficient adjustment using SOS constraints. *Operations Research* **49**(1) 175-177.

Magazine, M. J., M. S. Chern. 1984. A note on approximation schemes for multidimensional knapsack problems. *Mathematics of Operations Research* **9** 244-247.

Magazine, M. J., O. Oguz. 1984. A Heuristic Algorithm for the multidimensional zero-one knapsack problem. *European Journal of Operational Research* **16** 319-326.

Mainwaring, A., J. Polastre, R. Szewczyk, D. Culler, J. Anderson. 2002. Wireless sensor networks for habitat monitoring. *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, Atlanta, Georgia, 88-97.

Manne, A. S., H. M. Markowitz. 1957. On the solution of discrete programming problems. *Econometrica* **25** 84-110.

Maritime Transportation Security Act. 2008. Retrieved September 25, 2008, http://www.tsa.gov/assets/pdf/MTSA.pdf.

Marsten, R. E., T. L. Morin. 1977. Optimal solutions found for Senju and Toyoda'0/1 integer programming problems. *Management Science* **23** 1364-1365.

Martello, S., D. Pisinger, P. Toth. 1999. Dynamic programming and strong bounds for the 0-1 knapsack problem. *Management Science* **45**, 414-424.

Martello, S., P. Toth. 1990. *Knapsack Problems.* Wiley, New york.

Morton, D. P., F. Pan, K. J. Saeger. 2007. Models for nuclear smuggling interdiction. *IIE Transactions* **39**(1) 3-14.

Moser, M., D. P. Jokanovic, N. Shiratori. 1997. An algorithm for the multidimensional multiple-choice knapsack problem. *IEECE Transactions Fundament Electron Devices* **80**(3) 582-589.

Nauss, R. M. 1978. The 0-1 knapsack problem with multiple choice constraints. *European Journal of Operational Research* **2** 125-131.

Nemhauser, G. L., P. H. Vance. 1994. Lifted cover facets of the 0-1 knapsack polytope with GUB constraints. *Operations Research Letters* **16** 225-263.

Nemhauser, G. L., L. A. Wolsey. 1988. *Integer and Combinatorial Optimization*. Wiley, New York.

Night Vision Camera. 2008. Retrieved September 25, 2008, http://www.nightvision.com.

Osorio, M. A., F. Glover, P. Hammer. 2002. Cutting and surrogate constraint analysis for improved multidimensional knapsack solutions. *Annals of Operations Research* **117** 71-93.

Padberg, M. 1980. (1, *k*)-configurations and facets for packing problems. *Mathematical Programming* **18** 94-99.

Pandit, R., P. M. Ferreira. 1992. Determination of minimum number of sensors and their locations for an automated facility - an algorithmic approach. *European Journal of Operational Research* **63**(2) 231-239.

Papadimitriou, C. 2005. Pareto optimal sensor locations for structural identification. *Computer Methods in Applied Mechanics and Engineering* **194** 1655-1673.

Park, J. H., G. Friedman, M. Jones. 2004. Geographical feature sensitive sensor placement. *Journal of Parallel and Distributed Computing* **64** 815-825.

Parra-Hernandez, R., N. J. Dimopolous. 2005. A new heuristic for solving the multi-choice multidimensional knapsack problem. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* **35**(5) 708-717.

Petersen, C. C. 1967. Computational experience with variants of the Balas algorithm applied to the selection of R&D projects. *Management Science* **13** 736-750.

Pirkul, H. 1987. A heuristic solution procedure for the multicontraint zero-one knapsack problem. *Naval Research Logistics* **34** 161-172.

Pisinger, D. 1995. A minimal algorithm for the multiple-choice knapsack problem. *European Journal of Operational Research* **83** 394-410.

Port of Houston. 2008. Retrieved September 25, 2008, http://www.portofhouston.com.

Ralphs, T. K., M. V. Galati. 2006. Decomposition and dynamic cut generation in integer linear programming. *Mathematical Programming* **106** 261-285.

Saaty, T. L. 1980. *The Analytic Hierarchy Process*. McGraw Hill, New York.

Sbihi, A. 2007. A best first search exact algorithm for the multiple-choice multi-dimensional knapsack problem. *Journal of Combinatorial Optimization* **13** 337-351.

Sherali, H. D., F. Glover. 2008. Higher-order cover cuts from zero-one knapsack constraints augmented by two-sided bounding inequalities. *Discrete Optimization* **5** 270-289.

Sherali, H. D., Y. Lee. 1995. Sequential and simultaneous liftings of minimal cover inequalities for GUB constrained knapsack polytopes. *SIAM Journal of Discrete Mathematics* **8**(1) 133-153.

Shih, W. 1979. Branch and bound method for the multi-constraint zero-one knapsack problem. *Journal of the Operational Research Society* **30**(4) 369-378.

Soyster, A. L., B. Lev, W. Slivka. 1978. Zero–one programming with many variables and few constraints. *European Journal of Operational Research* **2** 195-201.

Stefan, B., Y. Nicola, F. Arnaud, A. Rumen. 2008. A dynamic programming based reduction procedure for the multidimensional 0-1 knapsack problem. *European Journal of Operational* **186**(1) 63-76.

Thermal Camera. 2008. Retrieved September 25, 2008, http://en.wikipedia.org/wiki/Infrared_camera.

Thermal Imaging. 2008. Retrieved September 25, 2008, http://www.emxinc.com/WhatisThermal Imaging.html.

Toyoda, Y. 1975. A simplified algorithm for obtaining approximate solutions to zero-one programming problems. *Management Science* **21**(12) 1417-1427.

Volgenant, A., J. A. Zoon. 1990. An improved heuristic for multidimensional 0-1 knapsack problems. *Journal of Operational Research Society* **41** 963-970.

Wang, J., N. Zhong. 2006. Efficient point coverage in wireless sensor networks. *Journal of Combinatorial Optimization* **11**(3) 291-304.

Weingartner, H. M., D. N. Ness. 1967. Method for the solution for the multidimensional 0–1 knapsack problem. *Operations Research* **15** 83-103.

Weistmental, R. 1997. On the 0-1 knapsack polytope. *Mathematical Programming* **77** 49-68.

Wilhelm, W. E. 2001. A technical review of column generation in integer programming, *Optimization and Engineering* **2** 159-200.

Wolsey, L. A. 1975. Faces for a linear inequality in 0-1 variables. *Mathematical Programming* **8** 165-178.

Wolsey, L. A. 1990. Valid inequalities for 0-1 knapsacks and MIPs with generalized upper bound constraints, *Discrete Applied Mathematics* **29** 251-261.

Zemel, E. 1978. Lifting facets of zero-one polytopes. *Mathematical Programming* **15** 268-277.

Zemel, E. 1989. Easily computable facets of the knapsack polytope. *Mathematics of Operations Research* **14** 760-765.

Zeng, B., J. P. Richard. 2006. Sequence independent lifting for 0–1 knapsack problems with disjoint cardinality constraints. Working paper, School of In-dustrial Engineering, Purdue University, West Lafayette, IN.

Zou, Y., K. Chakrabarty. 2005. A distributed coverage and connectivity centric technique for selecting active nodes in wireless sensor networks. *IEEE Transactions on Computers* **54**(8) 978-991.

VITA

Name: Elif İlke Çimren

Address: Department of Industrial and Systems Engineering
Texas A&M University
College Station, TX 77843-3131

Email Address: elifg@tamu.edu

Education: B.S., Istanbul Technical University, Turkey, 2002
M.S., Sabanci University, Turkey, 2004
Ph.D., Texas A&M University, TX, 2009