# SIMULTANEOUSLY LIFTING MULTIPLE SETS IN BINARY KNAPSACK INTEGER PROGRAMS

by

LAUREN ASHLEY KUBIK

B.S., Kansas State University, 2009

A THESIS

Submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Industrial and Manufacturing Systems Engineering

College of Engineering

KANSAS STATE UNIVERSITY

Manhattan, Kansas

2009

Approved by:

Major Professor

Dr. Todd Easton

# ABSTRACT

Integer programs (IPs) are mathematical models that can provide organizations with the ability to optimally obtain their goals through appropriate utilization and allocation of available resources. Unfortunately, IPs are $\mathcal{NP}$-complete in the strong sense, and many integer programs cannot be solved.

Introduced by Gomory, lifting is a technique that takes a valid inequality and strengthens it. Lifting can result in facet defining inequalities, which are the theoretically strongest inequalities; because of this, lifting techniques are commonly used in commercial IP software to reduce the time required to solve an IP.

This thesis introduces two new algorithms for exact simultaneous up lifting multiple sets into binary knapsack problems and introduces sequential simultaneous lifting. The Dynamic Programming Multiple Lifting Set Algorithm (DPMLSA) is a pseudo-polynomial time algorithm bounded by $O(nb)$ effort that can exactly uplift an arbitrary number of sets. The Three Set Simultaneous Lifting Algorithm (TSSLA) is a polynomial time algorithm bounded by $O(n^2)$ and can exact simultaneously up lift three sets. The simultaneously lifted inequalities generated by the DPMLSA and the TSSLA can be facet defining, and neither algorithm requires starting with a minimal cover.

A brief computational study shows that the DPMLSA is fast and required an average of only 0.070 seconds. The computational study also shows these sequential simultaneously lifted inequalities are useful, as the solution time decreased by an overall average of 18.4%. Therefore, implementing the DPMLSA should be beneficial for large IPs.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Integer programming has been successfully applied to minimize costs and manage resources in many industries. Integer programs (IPs) are mathematical models that can provide organizations with the ability to optimally obtain their goals through appropriate utilization and allocation of available resources.

One successful IP application by Delta Airlines saved approximately 100 million dollars per year by implementing optimal fleet assignments. More than 2,500 domestic flights and 450 airplanes per day are assigned by this IP [65]. Similarly, American Airlines saves over 20 million dollars per year using an IP to determine the crew schedule every month [1].

Many other successful IP applications have saved companies millions of dollars or benefited society. For instance, IPs have been applied in many areas to obtain optimal decisions, from researching genetics [14, 29] and fighting cancer [52, 53], to managing

portfolios [12, 59], transporting goods [2, 48, 61, 67], and developing sports schedules [23, 68].

## 1.1 Integer Programming

As represented by the examples above, integer programming is a tool used to solve various problems. Individuals can easily create a model to solve for optimal decisions. Unfortunately, IPs are $\mathcal{NP}$-complete in the strong sense [47], and even considering substantial scientific and computational advancements, many integer programs cannot be solved. Many times an IP may run for days without a solution being found. When this situation arises, practitioners must decide what concessions must be made for a solution. Many times the model must be altered and a non-optimal solution must be evaluated for its feasibility and benefit.

The simplest way to solve an IP is to enumerate all valid combinations and find the best value that satisfies the objective function. Though this enumeration provides the optimal integer solution, the larger the problem, the less feasible it is to list all possible answers. For every $n$ binary variables, there are $2^n$ possible combinations; clearly, the number of combinations is even larger when the variables are not binary, and frequently, this approach is not computationally viable.

The most common IP solution method is branch and bound [55]. This method is an exhaustive search approach, which finds the optimal solution, if one exists. Once an integer solution is found, the branch and bound method provides rules for disregarding

solution space, "branches", that will not provide a better answer.

Unfortunately, bad branching choices could lead again to the full $2^n$ solutions being enumerated. Obviously, this method is not computationally viable in such a situation. Therefore, both of these methods are exponential time solutions and not ideal for large problems. To help improve the solution time of IP solutions, cutting planes are found.

The cutting plane method was first introduced by Gomory [30]. Cutting planes are developed to limit the solution space by removing parts of the linear relaxation without disregarding any feasible integer solutions. The valid inequalities generated by the cutting plane method are useful only if they eliminate significant areas of the linear relaxation without eliminating any feasible integer point. One common technique to create useful cutting planes is lifting.

Lifting, first developed by Gomory [34], has been a significant area of research ever since. Lifting requires a starting valid inequality and then seeks to make the inequality stronger. This is done through one of the eight main lifting methods. This thesis focuses on a particular version of lifting called exact simultaneous up lifting. This type of lifting can result in facet defining inequalities, which are the theoretically strongest inequalities.

## 1.2 Research Motivation

Until recently, the only method to perform exact simultaneous up lifting was computationally intensive and required solving exponentially many integer programs. In 2005,

Hooker [44] developed a linear time method to exact simultaneously up lift a single set into a cover inequality for the knapsack polytope. In 2007, Gutierrez's [41] developed a method to simultaneously up lift multiple sets into an arbitrary IP, but it required the solution to a single integer program.

The motivation for this research is to intersect Gutierrez's method with Hooker's method. Thus, the goal at the outset of this research was to develop a fast or polynomial time method to simultaneously up lift a third set into a cover inequality for the knapsack polytope. If this could be achieved, then new facet defining results for the knapsack polytope could be quickly discovered.

## 1.3 Research Contributions

This thesis presents two new algorithms, a pseudo-polynomial time algorithm, the Dynamic Programming Multiple Lifting Set Algorithm (DPMLSA), and a polynomial time algorithm, Three Set Simultaneous Lifting Algorithm (TSSLA). These algorithms compute the exact simultaneously up lifted coefficient for multiple sets in binary knapsack problems.

The first algorithm, the DPMLSA, works with an arbitrary number of sets which are exact simultaneously up lifted to generate a new inequality. In certain instances, the inequality produced by the DPMLSA may be facet defining. The significant advantage presented by this thesis is that the DPMLSA only requires $O(nb)$ effort and is a pseudo-polynomial time algorithm. A small computational study shows that the DPMLSA is

fast and decreased the solution time to random knapsack instances by an average of 18.4%.

Similar to the DPMLSA, the TSSLA finds exact simultaneously up lifted coefficients. The TSSLA may also produce facet defining inequalities, in certain instances, and requires only quadratic effort. The TSSLA can be logically expanded for more than three sets, and the effort required for such an increase changes the running time by an order of $n$ for every additional set.

This research exceeded the initial goals of the project. The presented algorithms, the DPMLSA and the TSSLA, generate valid inequalities without requiring the input sets to be covers. Additionally, the DPMLSA can exactly simultaneously lift an arbitrary number of sets, in binary knapsack integer programs. From a theoretical view point, simultaneous up lifting over binary knapsack polyhedra is now completely understood. Fast algorithms and facet defining results exist due to this research. However, there still exists ample work in computational results and determining what sets should be simultaneously lifted as well as extensions to mixed integer programs.

## 1.4 Outline

Chapter 2 presents the basic concepts of polyhedral theory and integer programming, and contains background information necessary for understanding the research presented in this thesis. Some of the main topics covered in this chapter include the knapsack problem, polyhedral theory, cutting planes, cover inequalities, sequential and simultaneous

lifting, and facet defining inequalities. This information is formally defined and examples are provided for further understanding.

Chapter 3 contains the advancements of this research. Both the DPMLSA and the TSSLA are described. The facet defining results are presented in this chapter, as well. An example problem, which is used for both algorithms, provides a facet defining inequality, and consists of three sets that could each be defined as a minimal cover. Theorems and proofs pertaining to the algorithms can be found in Chapter 3, along with the step by step determination of the running time for each algorithm.

The computational results from the DPMLSA can be found in Chapter 4. This chapter presents the results and trials of the testing, along with an interpretation of the results. It also shows that simultaneously up lifting multiple sets are easy to find and extremely useful.

Chapter 5 summarizes this thesis. This chapter also contains ideas and extensions discovered during the development of the DPMLSA and the TSSLA that could be pursued as future research.

# Chapter 2

# Background Information

The core concepts of this research are better understood after a brief review of the fundamentals of integer programming, contained within this chapter. An interested reader can find many more details in [55].

An Integer Program (IP) contains an objective equation that must be minimized or maximized and a set of constraints which defines the solution space. An IP follows the form maximize $c^T x$, subject to $Ax \leq b$, where $x \in \mathbf{Z^n}$, $c \in \Re^{\mathbf{n}}$, $A \in \Re^{\mathbf{m \times n}}$ and $b \in \Re^{\mathbf{m}}$.

This chapter provides information necessary to understand the advancements of this research. The general fundamentals of polyhedral theory, including convex sets, half-spaces, polyhedrons and polytopes, linear relaxations, dimension, and affine independence, are discussed. These concepts of polyhedral theory combine to explain the usefulness of cutting planes, which are discussed in depth. The famous cutting planes, cover inequalities, and their relationship to the knapsack integer program are explained.

Finally, a broad survey of the different lifting methods is presented.

## 2.1 Polyhedral Theory

The fundamental concepts of many optimization methods stem from polyhedral theory. Polyhedral theory encompasses the feasible sets of linear programming problems. The relevant definitions from polyhedral theory are discussed in this section, and further information can be found in [55].

A set is a convex set if every point on the line segment connecting any two points from the set is also in the set. Formally, $S$ is convex if, and only if, $\lambda s_1 + (1-\lambda)s_2 \in S$ for all $s_1,\ s_2 \in S$ and $\lambda \in [0,1]$. The convex hull of a set $S$ is the intersection of all convex sets containing $S$.

The solution space for a single linear inequality is known as a halfspace, i.e. all $x \in \Re^{\mathbf{n}}$ such that $\Sigma_{j=1}^{n} a_j x_j \leq b$. A halfspace is clearly convex. A polyhedron is the intersection of a finite number of halfspaces. Therefore, the feasible region of a linear program $\{x \in \Re_{+}^{\mathbf{n}} : Ax \leq b\}$ is a polyhedron and is also convex. A bounded polyhedron is known as a polytope.

Define the set of feasible solutions to an integer program to be $P$. Equivalently, $P = \{x \in \mathbf{Z}_{+}^{\mathbf{n}} : Ax \leq b\}$. The solution space, $P$, consists of a countable set of points, is not a continuous region and is therefore not convex. Fundamentally important to cutting plane research is the convex hull of $P$, which is called $P^{ch} = conv(P)$. Observe

that $P^{ch}$ is a polyhedron.

Most of the methods employed to solve integer programs depend on the linear relaxation of the problem. The linear relaxation of an integer program is found by removing the integer constraint from the IP. That is, given an IP of the form maximize $c^T x$, subject to $Ax \leq b,\ x \in \mathbf{Z}^{\mathbf{n}}_{+}$, the corresponding linear relaxation denoted by $\text{IP}^{LR}$ is maximize $c^T x$, subject to $Ax \leq b, x \in \mathbf{R}^{\mathbf{n}}_{+}$. Define $P^{LR}$ to be the polyhedron of the feasible solution space for the linear relaxation, $P^{LR} = \{x \in \mathbf{R}^{\mathbf{n}}_{+} : Ax \leq b\}$.

The dimension of a polyhedron is typically defined as the number of linearly independent vectors contained in a polyhedron. The dimension of the polyhedron is critical to determining $P^{ch}$ and cutting planes (defined in the next section). Considering that $P^{ch}$ is derived from a collection of points, affine independence is required. A collection of points $x_1, x_2, x_3, ..., x_r \in \mathbf{R}^n_+$ are affinely independent if, and only if, $\Sigma^r_{j=1}\lambda_j x_j = 0$ and $\Sigma^r_{j=1}\lambda_j = 0$ is uniquely solved by $\lambda_j = 0\ \forall j = 1, ..., r$.

Affine independence is used to determine the dimension of the convex hull of a set of discrete points. The dimension of such a convex set is calculated as the maximum number of affinely independent points minus one. The subtraction of one is due to the idea that one of these points represents the origin and linearly independent vectors can be produced from this origin to the other affinely independent points.

### 2.1.1 Cutting Planes

Now that a polyhedron and its dimension have been defined, the concepts of cutting planes, valid inequalities, faces, and facets can be introduced. This section covers the concept of cutting planes and their importance to integer programming.

The purpose of a cutting plane or valid inequality is to eliminate a portion of $P^{LR}$ without eliminating any points in $P$ (a feasible integer solution). Thus, a cutting plane removes non-integer solution space from consideration by the linear relaxation. Formally, an inequality $\Sigma_{j=1}^{n}\alpha_j x_j \leq \beta$ is valid for $P^{ch}$ if, and only if $\Sigma_{j=1}^{n}\alpha_j x'_j \leq \beta$ is satisfied for every $x' \in P$.

Every valid inequality defines a face of a polyhedron. The face is the set of all points in the polyhedron that meet the inequality at equality. Thus, the valid inequality $\Sigma_{j=1}^{n}\alpha_j x_j \leq \beta$ defines a face $F \subseteq P^{ch}$ of the form $F = \{x \in P^{ch} : \Sigma_{j=1}^{n}\alpha_j x_j = \beta\}$. If $F \neq \emptyset$, then $F$ is said to support $P^{ch}$.

When considering $P^{ch}$ the only inequalities necessary to describe $P^{ch}$ are the facet defining inequalities. Facet defining inequalities have corresponding faces of $P^{ch}$ with dimension exactly one less than the dimension of $P^{ch}$. These faces of $P^{ch}$ are known as facets of $P^{ch}$.

Facet defining inequalities are important to researchers due to the fact that they are both necessary and sufficient for the description of $P^{ch}$. Thus, any inequality that is not a facet defining inequality of $P^{ch}$ is redundant and not necessary in its description. Determining $P^{ch}$ is critical to integer programming research, because solving a linear

program over $P^{ch}$ generates an optimal integer solution. Therefore, branch and bound need not be applied and the optimal solution can be generated at the root node.

The following example depicts these concepts.

**Example 1**

Consider the following integer program:

Maximize

$x_1 + 2x_2$

Subject to

$3x_1 + 2x_2 \leq 12$

$3x_1 + 4x_2 \leq 15$

$x_1,\ x_2\ \in \mathbf{Z}_+$.

Figure 2.1 provides a graphical view of this IP. The first constraint, $3x_1 + 2x_2 \leq 12$, passes through points $(0, 6)$, $C$, and $D$. The second constraint, $3x_1 + 4x_2 \leq 15$, passes through the points $A$, $B$, $C$, and $(5, 0)$. The linear relaxation of the IP is defined by these two constraints, the $x_1$ axis, and $x_2$ axis. The large circles represent the feasible integer points of the problem.

Looking at the graph, clearly there is space within the linear relaxation that is outside the feasible integer points. The aim of a cutting plane is to remove this non-integer solution space without eliminating any of the feasible integer points. The dashed line represents the valid inequality $x_1 + x_2 \leq 4$ and passes through the points $(0, 4)$, $B$, and $D$. This cutting plane eliminates the region $BCD$ of the linear relaxation space without cutting off any feasible integer points. Thus, it is a valid inequality.

By finding the dimension of $P^{ch}$ and the dimension of the cut's faces, this cutting

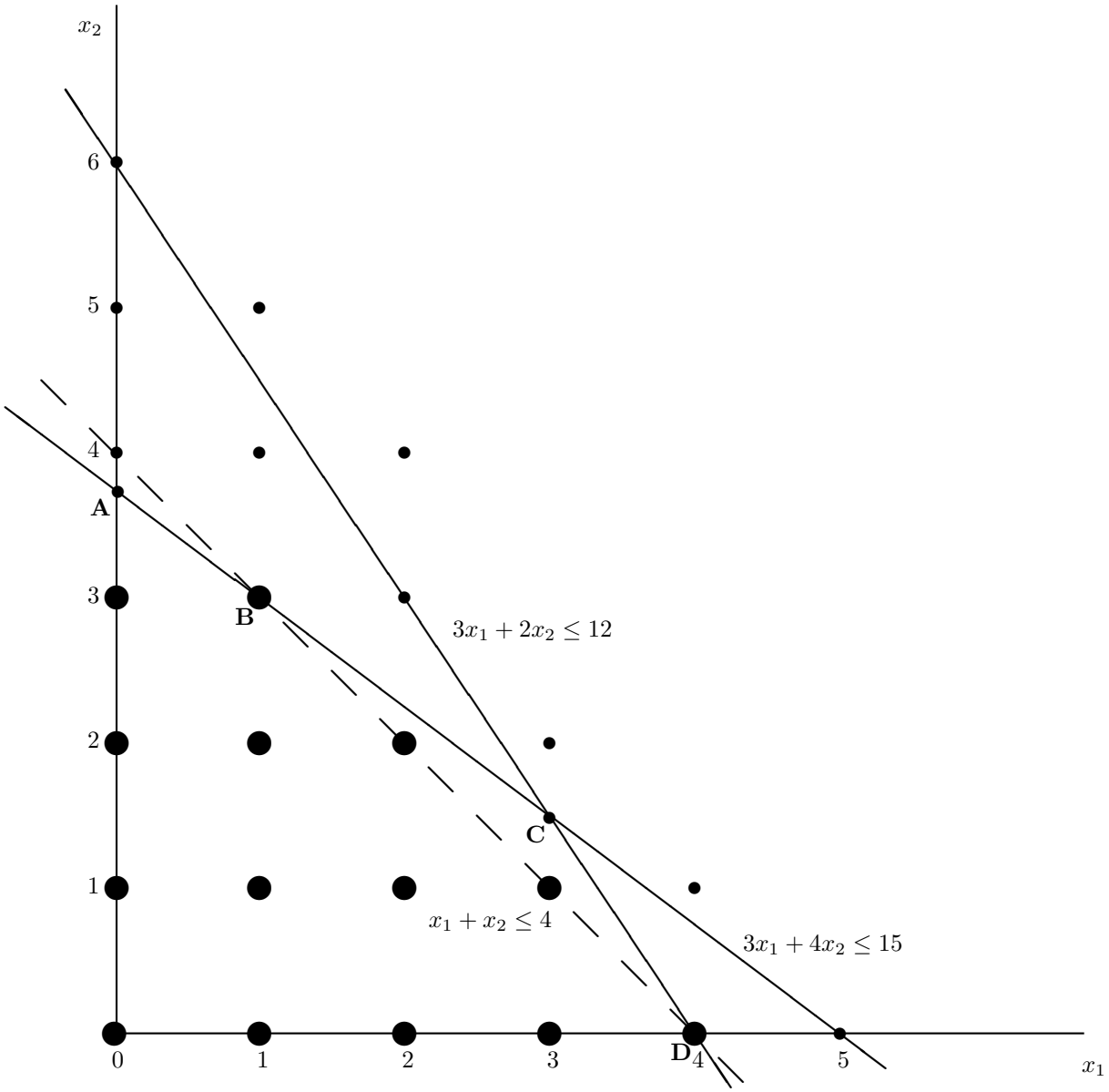


Figure 2.1: Cutting Plane Method in Example 1

plane can be classified as facet defining. The dimension of the polyhedron, $P^{ch}$, defined by the IP, is 2, which is found by listing three affinely independent points $(0,0)$, $(0,1)$, and $(1,0)$. Next the dimension of the cut's face is found by listing affinely independent points that meet the inequality at equality; in this case, $B$ and $D$ can be used as the points.

For this example, the other facet defining inequalities are $x_1 \geq 0$, $x_2 \geq 0$, and $x_2 \leq 3$. If these inequalities are added, then the reader can see $P^{ch}$, notice that all corner points are integer.

This example shows the simplicity of finding the valid inequalities in a two dimension integer programming problem. As the number of variables increases, so does the complexity of finding valid inequalities and facet defining inequalities. Knapsack problems are the simplest IPs with $n$ variables and are the focus of the next section and this thesis.

## 2.2 Knapsacks and Covers

Of particular importance to integer programming research is the Knapsack Problem (KP), a particular class of integer programs. The title "knapsack" refers to the analogy of a camper packing a knapsack with the items needed for an outing. There are $n$ items available for consideration and each has an associated benefit, $c_j$, and an associated weight, $a_j$ which is limited by the overall weight constraint of $b$. The camper seeks the items that give the most benefit while being restricted by the amount of weight he/she can carry. Therefore, the camper must find the maximum benefit while still being able

to carry the knapsack.

For this research, the binary form of the KP is considered; the camper can decide to take the item, $x_j = 1$, or go camping without it, $x_j = 0$. Therefore, an IP formulation for the binary KP is as follows maximize $\Sigma_{j=1}^{n} c_j x_j$, subject to $\Sigma_{j=1}^{n} a_j x_j \leq b$, $x_j \in \mathbf{B}$ for all $j = 1, 2, ..., n$. For the binary knapsack problem, $PKP$ represents the set of feasible solutions, and the convex hull of $PKP = \{x \in \mathbf{B}^n \ : \ \Sigma_{j=1}^{n} a_j x_j\}$ is represented by $PKP^{ch}$, which is a polyhedron.

The following example is used throughout this thesis and helps demonstrate this KP formulation.

**Example 2**

Consider a camper debating taking 13 objects on a trip. The benefit and weight of each item is in Table 2.1. For instance, $x_1$ is a tent, observe that the benefit of bringing the heaviest item is small, if it is a clear summer night. Similarly, $x_{13}$ represents a box of matches, the benefit of the lightest item is large, which indicates that having a campfire and warm meal are important to the camper.

| Object | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Benefit | 2 | 4 | 8 | 4 | 6 | 8 | 11 | 7 | 9 | 10 | 8 | 9 | 12 |
| Weight | 37 | 36 | 36 | 35 | 34 | 23 | 23 | 22 | 22 | 21 | 21 | 20 | 20 |

Table 2.1: Benefit and Weight of Objects

The objective maximizes the expected benefit, and the constraint enables the camper to carry the weight. It is now straightforward to create an IP to solve this problem. Let

the decision variable be $x_j = 1$ if the $j^{th}$ item is taken, and $x_j = 0$ if not. The IP formulation is

Maximize

$$2x_1 + 4x_2 + 8x_3 + 4x_4 + 6x_5 + 8x_6 + 11x_7 + 7x_8 + 9x_9 + 10x_{10} + 8x_{11} + 9x_{12} + 12x_{13}$$

Subject to

$$37x_1 + 36x_2 + 36x_3 + 35x_4 + 34x_5 + 23x_6 + 23x_7 + 22x_8 + 22x_9 + 21x_{10} + 21x_{11} + 20x_{12} + 20x_{13} \leq 152$$

$$x_j = \{0, 1\} \; \forall j = 1, ..., 13 \; .$$

Much theoretical research has been done on cutting planes for the knapsack polyhedron [4, 6, 7, 20, 45, 55, 56, 58, 63, 73]. The reason that the knapsack polyhedron is of particular interest to IP researchers is that any single IP constraint can be modified to become a KP. This transformation is done by changing any constraints of the '=' or '$\geq$' type to the '$\leq$' type. An '=' equation can be split into two constraints, one with '$\geq$' and one with '$\leq$'. The '$\geq$' constraints can then be multiplied by $-1$ to form '$\leq$' constraints. If any coefficient $a_j < 0$, then substitute $(1 - x'_j)$ for $x_j$; the resulting inequality has only positive coefficients. Therefore, any single binary IP constraint is a equivalent to a knapsack constraint.

The most relevant work to this thesis involves covers of knapsack constraints. Valid inequalities known as cover inequalities can be developed from the original knapsack constraints to further define the solution space. A cover inequality is developed by taking a subset of items that are too heavy to carry and then define a constraint to limit the number of these items taken.

Formally, given a KP, a cover is any set $C \subseteq N$ such that $\Sigma_{j \in C} a_j > b$. That is, the set $C$ represents a set of points that is infeasible. Every cover defines a cover inequality,

of the form $\Sigma_{j \in C} x_j \leq |C| - 1$ in $PKP^{ch}$. Clearly, every cover inequality is valid.

A cover is minimal if $\Sigma_{j \in C \setminus \{l\}} a_j \leq b$ for each $l \in C$. Minimal covers always produce inequalities of dimension at least $|C| - 1$ in $PKP^{ch}$. This can be seen by taking the points $\Sigma_{j \in C \setminus \{l\}} e_j$ for each $l \in C$ where $e_j$ is the $j^{\text{th}}$ identity point.

An extended cover is formally defined as $E(C) = C \cup \{i \in N - C : a_i \geq max_{j \in C}\{a_j\}\}$. An extended cover is generated by taking the cover and adding the indices with coefficients greater than or equal to the largest coefficient in $C$. Clearly, an extended cover inequality, $\Sigma_{j \in E(C)} x_j \leq |C| - 1$, is also a valid inequality.

Returning to the knapsack instance from Example 2, observe that a cover is $\{6, 7, ..., 13\}$, because the sum of the constraint coefficients is $20+20+21+21+22+22+23+23 = 172$, which is greater than 152. Logically, because the sum of the coefficients of the eight variables is greater than 152, they cannot all be selected at the same time. This logic provides us with the cover inequality $\Sigma_{j=6}^{13} x_j \leq 8 - 1 = 7$. This is a minimal cover, because if any element of the cover is removed, the sum of the other 7 coefficients is less than or equal to 152 and feasible.

The cover inequality for this example defines a face of dimension 7. The eight affinely independent points are generated by setting each variable corresponding to an index in the cover to zero, one at a time, with the other seven set to one. These points are feasible and meet the cover inequality at equality. For example, when $x_{13}$ (the smallest coefficient) is set to zero adding all of the coefficients for $x_6$ through $x_{12}$ is less than or equal to 152. Thus, the point $(0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0)$ is feasible and meets the

cover inequality at equality. Since this is true for $x_{13} = 0$ (the smallest coefficient), setting any of the other indices to zero produces a similar result. The following points are listed in matrix notation with the points being listed as columns. The set of 8 affinely independent points show that this cover inequality defines a face of dimension at least 7 over $PKP^{ch}$

$$\begin{matrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0
\end{matrix} .$$

Even though this is a valid inequality defining a face of dimension 7, it can be improved. The current cover inequality only considers eight of the thirteen variables from the objective function and original constraint. The rest of the variables could be added to the cover using the concept of an extended cover. The extended cover for this problem would be $\Sigma_{j=1}^{13} x_j \leq 7$. This inequality is valid because all coefficients from $x_1$ through $x_5$, from the original inequality, are greater than 23, the largest associated coefficient for the minimal cover. Though it is valid, the extended cover inequality still defines a face of dimension 7, and is therefore not facet defining over $PKP^{ch}$.

Clearly, the goal is to create an inequality of dimension 12, which would be facet defining for $PKP^{ch}$. It is possible that this minimal cover inequality may be modified to become facet defining; if so, the coefficients of $x_1$ through $x_5$ must be changed. Lifting is a technique that changes coefficients and can increase the dimension of the face defined by the inequality.

## 2.3 Lifting

First introduced by Gomory [34], lifting is a common method used to increase the dimension of a cut. Lifting takes a valid inequality and, by changing some of the coefficients and possibly the right hand side, makes the inequality stronger. Lifting is also used to determine cutting planes with potential to be facet-defining inequalities. Many arcticles present this topic [4, 6, 7, 8, 9, 10, 15, 18, 20, 21, 28, 38, 39, 50, 56, 58, 60, 72, 73].

The idea of a restricted space is fundamental to lifting. Define the restricted space of $P^{ch}$ on the set of $E$ variables to be $P^{ch}_{E,K} = conv\{x \in P : x_j = k_j \text{ for all } j \in N \setminus E\}$ where $k_j \in \mathbf{Z}$ and $K = \{k_1, k_2, ..., k_{|N \setminus E|}\}$. Thus, rather than looking at the entire polyhedron, the restricted space considers only a subset of the variables. In other words, every variable with an index in $E$ is forced to take on a fixed value, denoted by $k_j$. This thesis focuses on what is known as up lifting, which requires $K = \{0, 0, ..., 0\}$. To simplify notation, define $P^{ch}_E$ to be $P^{ch}_{E,K}$ whenever $K = \{0, 0, ..., 0\}$.

An inequality can be classified as facet defining over the restricted space if all of the facet defining requirements are satisfied when only considering the restricted space.

Once an inequality is facet defining on a restricted space, researchers try to maintain the facet defining property while changing the restricted space to the entire space.

Returning to Example 2, the minimal cover is facet defining over the restricted space defined by $x_6$ through $x_{13}$. The dimension of the inequality would remain 7 which would be one less than the dimension of the restricted space. The eight affinely independent points defining the dimension (listed vertically) would be as follows:

$$\begin{array}{cccccccc}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 0
\end{array} .$$

With lifting, the goal clearly is to modify a valid inequality to make it facet defining over $P^{ch}$. By starting with a restricted space facet defining inequality, lifting modifies the inequality to create an inequality, which may be facet defining for $P^{ch}$.

There are multiple types of lifting [6, 7, 42, 57, 69, 70, 73] such as sequential, simultaneous, exact, approximate, up and down lifting techniques. Therefore, there are at least eight different types of lifting and include exact sequential up lifting, exact sequential down lifting, approximate sequential up lifting, approximate sequential down lifting, exact simultaneous up lifting, exact simultaneous down lifting, approximate simultaneous

up lifting, and approximate simultaneous down lifting.

Formally, let $\Sigma_{j \in N \setminus E} \alpha_j x_j \leq \beta$ be a valid inequality of $P^{ch}_{E,K}$, then lifting seeks to create a valid inequality of $P^{ch}$ that takes the form $\Sigma_{j \in E} \alpha'_j x_j + \Sigma_{j \in N \setminus E} \alpha'_j x_j \leq \beta'$. The values of $\alpha'$, $\beta'$ and $E$ determine the aforementioned types of lifting.

Exact lifting requires that all coefficients are as strong as possible. Thus, any increase in a lifted coefficient on the left hand side or a decrease in the right hand side results in an invalid inequality. Typically, exact lifting requires the solution to some optimization problem, which is typically an integer program. Thus, exact lifting can require too much computational effort to be effective, although some individuals have developed polynomial time algorithms to perform exact lifting on the knapsack polytope [8, 26, 63].

Since solving optimization problems can be computationally intensive, many researchers have provided results on approximate lifting. Approximate lifting seeks to remove the computational complexity associated with exact lifting. So rather than find the best possible lifting coefficients in a long time, a fast heuristic will generate coefficients that maintain a valid inequality, but these coefficients may be able to be improved. Some approximate lifting results include sequential lifting [6] and sequence independent lifting [5, 40, 64, 71], which can be considered an approximate simultaneous lifting technique. Other approximation work has used a linear relaxation or just a portion of the original problem to approximate the lifting coefficients [62, 70].

Up versus down lifting refers to the values of $K$ and $\beta'$. For up lifting, it is assumed that every element in $K$ is equal to zero. In such a case, $\beta' = \beta$. Down lifting considers

each element of $K$ to be equal to its upper bound. Down lifting usually decreases the right hand side of the inequality. There is also middle lifting, which considers $K$ values to be between zero and the upper bound and is a combination of up and down lifting.

Sequential and simultaneous lifting are distinguished by the size of $E$. Sequential lifting assumes $|E| = 1$, while simultaneous lifting has $|E| > 1$. Thus, a sequentially up lifted inequality assumes $\Sigma_{j=2}^{n} \alpha_j x_j \leq \beta$ is valid for $P^{ch}_{\{1\},\{0\}}$ and seeks an inequality of the form $\alpha_1 x_1 + \Sigma_{j=2}^{n} \alpha_j x_j \leq \beta$ that is valid for $P^{ch}$. Typically, sequential lifting is an exact lifting technique, which seeks to find the strongest $\alpha_1$ possible. Therefore, exact sequential up lifting finds the maximum value of $\alpha_1$ that still maintains the validity of the inequality. If such a value of $\alpha_1$ is obtained, then the dimension of the face induced by the sequentially lifted inequality typically increases by at least 1 when moving from $P^{ch}_{\{1\},\{0\}}$ to $P^{ch}$.

Simultaneous lifting considers the entire set of $E$ variables when determining the coefficients and requires $|E| \geq 2$. The most common type of simultaneous lifting assumes that $\alpha_j = \alpha$ for every $j \in E$. However, other versions are possible. The main idea behind simultaneous lifting is to lift many variables at the same time rather than lifting one variable at a time.

### 2.3.1 Sequential Lifting

Of the types of lifting, sequential lifting is the most widely studied and used. Sequential lifting adds a variable to the original valid inequality individually and determines the

coefficient necessary for that variable. Sequential lifting requires $|E| = 1$; therefore, sequential up lifting assumes that $\Sigma_{j=2}^{n}\alpha_j x_j \leq \beta$ is valid for $P_{\{1\}}^{ch}$ and seeks to create a valid inequality for $P^{ch}$ of the form $\alpha_1 x_1 + \Sigma_{j=2}^{n}\alpha_j x_j \leq \beta$. By iteratively performing sequential lifting, researchers can start with a facet defining inequality of a restricted space and end with a facet defining inequality of the whole space as long as $P^{ch}$ is full dimensional.

Exact sequential up lifting of a binary variable, $x_j$, requires the solution to the following IP.

$$
\begin{aligned}
z^* = \text{Maximize } & \Sigma_{j=2}^{n}\alpha_j x_j \\
\text{Subject to: } & Ax \leq b \\
& x_1 = 1 \\
& x_1 \in \{0,1\},\ x_2, ..., x_n \in \mathbf{Z}^n
\end{aligned}
$$

Once the optimal solution is obtained, $\alpha_1 = \beta - z^*$.

Applying sequential lifting to Example 2 creates a facet defining inequality for $PKP^{ch}$. Recall that the original integer program is

Maximize

$$2x_1 + 4x_2 + 8x_3 + 4x_4 + 6x_5 + 8x_6 + 11x_7 + 7x_8 + 9x_9 + 10x_{10} + 8x_{11} + 9x_{12} + 12x_{13}$$

Subject to

$$37x_1 + 36x_2 + 36x_3 + 35x_4 + 34x_5 + 23x_6 + 23x_7 + 22x_8 + 22x_9 + 21x_{10} + 21x_{11} + 20x_{12} + 20x_{13} \leq 152$$

$$x_j = \{0,1\}\ \forall\ j = 1, ..., 13.$$

The valid cover inequality, $\Sigma_{j=6}^{13} x_j \leq 7$ is facet defining over $PKP_{\{6,7,...,13\}}^{ch}$. In the next few paragraphs, sequentially up lifting the other five variables is discussed.

To sequentially up lift $x_1$, solve the integer program: Maximize $\Sigma_{j=6}^{13} x_j$, Subject to

$37x_1 + 36x_2 + ... + 20x_{13} \leq 152$, $x_1 = 1$, and $x$ in $\{0, 1\}$ $\forall$ $j = 2, ..., 13$. The value for the objective $z^* = 5$. Therefore, $\alpha_1 = 7 - 5 = 2$ and the new sequentially up lifted inequality is $2x_1 + \Sigma_{j=6}^{13} x_j \leq 7$.

Next $x_2$ can be lifted in the same manner, using the new inequality, $2x_1 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} \leq 7$, as the objective function. The $z^* = 5$ and $\alpha_2 = 7 - 5 = 2$. Continuing for the rest of the variables, for $x_3$ the $z^* = 6$ and $\alpha_3 = 7 - 6 = 1$, for $x_4$ the $z^* = 6$ and $\alpha_4 = 7 - 6 = 1$, and for $x_5$ the $z^* = 6$ and $\alpha_5 = 7 - 6 = 1$. The final exact sequentially up lifted inequality is $2x_1 + 2x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} \leq 7$.

This inequality is facet defining for $PKP^{ch}$. The thirteen affinely independent points necessary to prove this are

$$
\begin{array}{cccccccccccccc}
1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\
1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0
\end{array}
.
$$

Balas [7] developed guidelines for both exact and approximate lifting coefficients. Theorem 2.3.1 provides his result.

**Theorem 2.3.1.** *Let $C = \{j_1, ..., j_r\}$ be a minimal dependent set with $j_1 < j_2 < ... < j_r$. Let $\mu_h = \Sigma_{k=1}^{h}\ a_{j_k}$ for $h = 1, ..., r$; also let $\mu_0 = 0$ and $\lambda = \mu_r - b \leq 1$. Every valid inequality of the form*

$$\Sigma_{j \in N \setminus C}\ \alpha_j x_j + \Sigma_{j \in C}\ x_j \leq |C| - 1$$

*that represents a facet of $conv(S)$ satisfies the following conditions:*

*i. If $\mu_h \leq a_j \leq \mu_{h+1} - \lambda$, then $\alpha_j = h$.*

*ii. If $\mu_{h+1} - \lambda + 1 \leq a_j \leq \mu_{h+1}$, then*

*(a) $\alpha_j \in [h, h+1]$ and*

*(b) there is at least one facet of the form $\Sigma_{j \in N \setminus C}\ \alpha_j x_j + \Sigma_{j \in C}\ x_j \leq |C| - 1$ with $\alpha_j = h + 1$.*

Observe that Balas' Theorem divides the coefficients into two distinct classes. The first is assigned a fixed integer coefficient and the second has a coefficient within a specified range. This result is critical for the computational study of the DPMLSA and is revisited in Chapter 4.

Sequential lifting is order dependent; in the example above $x_1$ was considered first and the elements were added in order. There are $5! = 120$ different order combinations that could have been attempted. If the order was varied, the sequentially up lifted minimal cover inequality could result in only seven different inequalities. The resulting inequalities (including the example's) are

$$2x_1 + 2x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} \leq 7,$$

$$2x_1 + x_2 + 2x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} \leq 7,$$
$$2x_1 + x_2 + x_3 + 2x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} \leq 7,$$
$$2x_1 + x_2 + x_3 + x_4 + 2x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} \leq 7,$$
$$x_1 + 2x_2 + 2x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} \leq 7,$$
$$x_1 + 2x_2 + x_3 + 2x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} \leq 7, \text{ and}$$
$$x_1 + x_2 + 2x_3 + 2x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} \leq 7.$$

Each of these inequalities is facet defining and necessary in the description of $PKP^{ch}$. Taking the average of the coefficients for each $x$ results in a single valid inequality. The inequality is $\frac{11}{7}x_1+\frac{10}{7}x_2+\frac{10}{7}x_3+\frac{10}{7}x_4+\frac{8}{7}x_5+x_6+x_7+x_8+x_9+x_{10}+x_{11}+x_{12}+x_{13} \leq 7$. The next section discusses simultaneous lifting and generates an inequality that dominates this average sequential up lifted inequality.

### 2.3.2 Simultaneous Lifting

Simultaneous lifting requires $|E| \geq 2$. Therefore, instead of performing sequential lifting $|E|$ times, all of the variables in $E$ can be lifted at one time. In 1978, Zemel [72] provided the first exact technique to simultaneously lift. Zemel's technique is limited to sets of binary variables and requires solving an exponential number of integer programs. Zemel's method works by finding the extreme points of the polar created from the solutions to these integer programs, and yields numerous inequalities. Clearly, Zemel's method is far too computationally intensive to ever be an effective tool.

Later on comments, but no algorithms, were provided by researchers regarding the existence and importance of exact simultaneous lifted inequalities [8, 39]. Gu, et al. [40] developed a linear time algorithm to perform approximate simultaneous up lifting,

which is called sequence independent lifting.

Recently, much work has been done at KSU on exact simultaneous lifting. Easton and Hooker developed a linear time algorithm to exactly simultaneously up lift sets of variables into a cover inequality for $PKP^{ch}$ [26]. Gutierrez's master's thesis presents an exact technique to simultaneously up lift sets of variables to general integer programs. Her method requires the solution to a single integer program. Sharma's thesis improved Easton and Hooker's work and could identify what sets should be lifted and improved the facet defining results. Her algorithm ran in quadratic time, generated many inequalities and had impressive computational results.

The premise of all of this work at KSU is to guess an inequality that may be invalid, but has the facet defining property. If the inequality is shown to be invalid, by the existence of a feasible point that violates this inequality, then the inequality is changed to make this point satisfy the inequality at equality. Gutierrez's method is presented here to demonstrate this concept.

The input to Gutierrez's method is a lifting set $E$ and a valid inequality $\Sigma_{j\in N\setminus E}\alpha_j x_j \leq \beta$ over $P_E^{ch}$. Her method can also take a weight function, but the simpler version is presented here. The exact simultaneously up lifted inequality takes the form $\alpha\Sigma_{j\in E}x_j + \Sigma_{j\in N\setminus E}\alpha_j x_j \leq \beta$. Her algorithm sets $\alpha = M >> 0$ and solves an integer program of the form maximize $\alpha\Sigma_{j\in E}x_j + \Sigma_{j\in N\setminus E}\alpha_j x_j \leq \beta$, subject to $Ax \leq b$, $x \in \mathbf{Z}^n$. The optimal solution to this problem is $Z^*$, $x^*$. If $Z^* > \beta$, then the next value of $\alpha$ becomes $\frac{\beta-\Sigma_{j\in N\setminus E}\alpha_j x_j^*}{\Sigma_{j\in E}x_j^*}$. Thus, the feasible point $x^*$ meets this new inequality at equality. This

process repeats until $Z^* \leq \beta$, then the inequality $\alpha\Sigma_{j\in E}x_j + \Sigma_{j\in N\setminus E}\alpha_j x_j \leq \beta$ is valid and $\alpha$ is reported as the lifting coefficient.

Returning to Example 2 demonstrates how Gutierrez applies this method to create simultaneously lifted inequalities. Recall that the primary example in this thesis is

$$37x_1+36x_2+36x_3+35x_4+34x_5+23x_6+23x_7+22x_8+22x_9+21x_{10}+21x_{11}+20x_{12}+20x_{13} \leq 152$$
$$x_j \ = \{0,1\} \forall\ j = 1, ..., 13.$$

The valid cover inequality, $\Sigma_{j=6}^{13}x_j \leq 7$ is facet defining over $PKP^{ch}_{\{1,2,3,4,5\}}$; the other five variables are now simultaneously up lifted.

To simultaneously lift $x_1$ through $x_5$, set $\alpha = M$ and solve the IP

Maximize
$\Sigma_{j=6}^{13}x_j + \Sigma_{j=1}^{5}Mx_j$
Subject to
$37x_1 + 36x_2 + ... + 20x_{13} \leq 152$
$x_j \ = \{0,1\}\ \forall\ j = 1, ..., 13$

The value for the objective, $z^* = 4M$, is obtained with $x^* = (0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0)$. Since $z^* > \beta$, the inequality $\Sigma_{j=6}^{13}x_j + \Sigma_{j=1}^{5}Mx_j \leq \beta$ is not valid due to $x^*$. A new $\alpha$ is obtained by plugging the $x^*$ value into the constraint and solving for $\alpha$ when the constraint is at equality. Thus, $\alpha = \frac{\beta - \Sigma_{j\in N\setminus E}\alpha_j x_j^*}{\Sigma_{j\in E}x_j^*}$, so $\alpha = \frac{7-0}{4} = \frac{7}{4}$.

To determine if this is the correct $\alpha$ value, the coefficient M is replaced by the new $\alpha$ value, $\frac{7}{4}$, and the IP is solved again. The objective value $z^* = (\frac{7}{4})(4) = 7$ is obtained with the same point, and $\frac{7}{4}x_1+\frac{7}{4}x_2+\frac{7}{4}x_3+\frac{7}{4}x_4+\frac{7}{4}x_5+x_6+x_7+x_8+x_9+x_{10}+x_{11}+x_{12}+x_{13} \leq 7$ is valid for $PKP^{ch}$ and is the exact simultaneously up lifted inequality. Notice that

this inequality dominates the average of the sequentially lifted inequalities, which is $\frac{11}{7}x_1 + \frac{10}{7}x_2 + \frac{10}{7}x_3 + \frac{10}{7}x_4 + \frac{8}{7}x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} \leq 7$.

Simultaneously up lifted inequalities may be facet defining, but are not guaranteed to be facet defining, as is the case for sequentially up lifted inequalities. In this example, the simultaneously up lifted inequality is facet defining. The thirteen affinely independent points that show this are

$$
\begin{array}{cccccccccccccc}
0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0
\end{array}
.
$$

The first five columns give the affinely independent points that were added by simultaneous lifting. These points increase the dimension from seven to twelve, making the simultaneously up lifted inequality facet defining. Observe that these five affinely independent points were generated using the point $x^*$ from above.

The purpose of this research was to extend the concepts of simultaneous up lifting to consider multiple sets in polynomial time for PKP instances. Thus, this research attempts to intersect Gutierrez's work with Hooker's work.

# Chapter 3

# Simultaneously Lifting Multiple Sets

This research stems from results by Gutierrez [41], Hooker[26], and Sharma[63]. The goal of this research was to develop fast methods to exact simultaneously up lift a third set into a cover inequality for the knapsack polytope. This initial goal was exceeded by the algorithms presented below. The DPMLSA algorithm provides a method for lifting an arbitrary number of sets, and both the DPMLSA and the TSSLA generate valid inequalities without requiring the sets to be covers. Furthermore, the TSSLA can be expanded to more sets, but the running time would increase by the order of $n$ with each added set.

This chapter provides an in depth explanation of both the pseudo-polynomial time algorithm, the DPMLSA, and the polynomial time algorithm, the TSSLA. Both algorithms generate exact simultaneously up lifted inequalities, and facet defining examples are provided after each algorithm. The supporting theorems are also presented in this

chapter.

## 3.1 Dynamic Programming Multiple Lifting Set Algorithm (DPMLSA)

The input to the DPMLSA is a knapsack constraint $\Sigma_{j=1}^{n} a_j x_j \leq b$, a lifting set $E_p \subset N$ and a valid inequality $\Sigma_{j \in N \setminus E_p} \alpha_j x_j \leq \beta$ over $PKP_{E_p}^{ch}$. For simplicity let $N \setminus E_p$ be partitioned into sets $E_1, E_2, ..., E_{p-1}$ such that $\Sigma_{j \in N \setminus E_p} \alpha_j x_j = \Sigma_{i=1}^{p-1} \alpha_i \ \Sigma_{j \in E_i} x_j$. In other words, each $E_i$ consists of all of the indices of the variables that have identical coefficients in the valid inequality.

The DPMLSA's basic idea is to create an array $T = (t_0, ..., t_b)$ of size $b+1$ and have $t_j$ represent the maximum that the left hand side of the valid inequality can be if the right hand side of the knapsack is changed to $j$. For simplicity of notation, $t_j < 0$ with $j < 0$ does not exist, and as such, the evaluated condition is never true.

Once $T$ is correctly calculated, the $k$ smallest $a$ values associated with indices in $E_p$ are subtracted from $b$ and set equal to $j$. Next, an upper bound on $\alpha$ is determined by subtracting $t_j$ from $\beta$ and then dividing by $k$. This step is repeated for all $k = 1, ..., |E_p|$ and the minimum $\alpha$ value is reported.

**Dynamic Programming Multiple Lifting Set Algorithm (DPMLSA)**

**Initialization**

Let $T$ be an array $T = (t_0, t_1, ..., t_b)$ with all elements set to 0.

*For* $i = 1$ *to* $p - 1$

Update $T$ by calling CreateTableValues ($T$, $E_i$).

*End For*

Let $F_p = \{f_1^p, ..., f_{|E_p|}^p\}$ where $F_p$ is $E_p$ sorted in ascending order based upon the corresponding $a$ values of the knapsack constraint.

$\alpha_p := \infty$.

**Main Step**

*For* $k = 1$ *to* $|E_p|$

*if* $(\beta - t_{b-\Sigma_{q=1}^k a_{f_q^p}})/k < \alpha_p$, *then* $\alpha_p = (\beta - t_{b-\Sigma_{q=1}^k a_{f_q^p}})/k$.

*End For*

**Output**

Report $\alpha_p^*$ as the exact simultaneously up lifted coefficient when $\alpha_p^* = \alpha_p$.

**Subroutine** CreateTableValues ($T$, $E_i$)

**Initialization**

$S := T$, $s_j = t_j$ for all $j = 0, ..., b$.

**Main Step**

Let $F_i = \{f_1^i, ..., f_{|E_i|}^i\}$ where $F_i$ is $E_i$ sorted in ascending order based upon the

corresponding $a$ values of the knapsack constraint.

$k := 1$.

*While* $k \leq |F_i|$

$j := 0$

*While* $j < b$

*if* $s_j < t_{j-\Sigma_{q=1}^{k} a_{f_q^i}} + \alpha_i k$, *then* $s_j = t_{j-\Sigma_{q=1}^{k} a_{f_q^i}} + \alpha_i k$.

j:=j+1.

*end While*

k:=k+1

*end While*

**Termination**

$T := S$, $t_j := s_j$ for all $j = 0, ..., b$.

Report $T$ as the updated table values.

To determine the running time of the DPMLSA, the algorithm is systematically evaluated. First, consider the subroutine $CreateTableValues$. The Initialization step takes $O(b)$ effort. Within the Main Step, sorting $E_i$ requires $O(|E_i|log(|E_i|))$ and the while loop clearly takes $O(|E_i|b)$ effort. The Termination requires $O(b)$. So, overall, the subroutine $CreateTableValues$ requires $O(|E_i|(b + log(|E_i|)))$ effort.

Returning to the DPMLSA portion of DPMLSA, Initialization requires $O(b)$. Using amortized analysis, the loop containing the $CreateTableValues$ subroutine becomes $O(\Sigma_{i=1}^{p-1}(|E_i|(b)+|E_i|log(|E_i|)))$. The final step in the initialization takes $O(|E_p|log(|E_p|)$. Clearly, $\Sigma_{i=1}^{p}|E_i|$ can be at most $O(n)$, and so, overall, the initialization requires at most $O(nb + nlog(n))$ effort.

The Main Step requires $O(|E_p|)$, and finally, the termination requires $O(1)$. Therefore, this algorithm, DPMLSA, runs in $O(nb + nlog(n))$ effort. Furthermore, it can be argued that $b > log(n)$ in most instances and the running time can be reasonably reported as $O(nb)$ effort.

The following theorem proves that correctly implementing the DPMLSA results in a constraint that is a valid inequality of $PKP^{ch}$. Thus, one can iteratively apply the DPMLSA numerous times until all variables have been simultaneously up lifted. Formally,

**Theorem 3.1.1.** *The inequality $\Sigma_{i=1}^{p}\alpha_i\Sigma_{j\in E_i}x_j \leq \beta$ is valid as long as $\alpha_p \leq \alpha_p^*$ where $\alpha_p^*$ is reported from the DPMLSA.*

*Proof:* For contradiction assume that $\Sigma_{i=1}^{p}\alpha_i\Sigma_{j\in E_i}x_j \leq \beta$ is not valid. Then there exists an $x'$ such that $\Sigma_{j=1}^{n}a_jx'_j \leq b$, $x' \in \{0,1\}$ and $\Sigma_{i=1}^{p}\alpha_i\Sigma_{j\in E_i}x'_j > \beta$ (†). Now define, $q_i = |\{x'_j = 1 : j \in E_i\}|$ for $i = 1, ..., p$. Observe that $q_p \geq 1$.

Let $s = \Sigma_{i=1}^{p}\Sigma_{j=1}^{q_i}a_{f_j^i}$ and $s' = \Sigma_{i=1}^{p-1}\Sigma_{j=1}^{q_i}a_{f_j^i}$. Clearly, $s' \leq s \leq b$. Since $t_j$ is monotonically nondecreasing through each step in the DPMLSA, $t_{s'} \geq \Sigma_{i=1}^{p-1}\alpha_i q_i$. Furthermore, $\alpha_p^*$ is nonincreasing and $t_{b-\Sigma_{j=1}^{q_p}a_{f_j}^p} \geq t_{s'}$. The main step calculates an upper bound on

$\alpha_p^*$ to be $\frac{\beta - t_{b-\Sigma_{q=1}^{k} a_{f_q^p}}}{q_p} \geq \frac{\beta - t_{s'}}{q_p}$. Thus, $\frac{\beta - t_{s'}}{q_p} q_p + t_{s'} = \beta$, which contradicts (†) and the result follows.

□

Therefore, the DPMLSA runs with $O(nb)$ effort, and generates the valid inequality $\Sigma_{i=1}^{p} \alpha_i \Sigma_{j \in E_i} x_j \leq \beta$. Furthermore, these inequalities may be facet defining, which is focus of the remainder of this section.

In order to generate some theoretical or facial defining results of these simultaneously lifted inequalities, some additional information must be recorded by the DPMLSA. For validity, the only thing that is important is whether or not there exists a feasible point that violates the inequality. For facial defining results, it is also important to know how many points meet an inequality at equality. The bases for these points will be referred to as $q'_{j_i}$ where $j = 0, ..., b$ and $i = 1, ..., p$ and can easily be obtained by modifying the DPMLSA.

The idea is to modify the $CreateTableValues$ subroutine by changing the 'then' condition, which comes after the 'if' condition ($if\, t_j < t_{j-\Sigma_{q=1}^{k} a_{f_q^i}} + \alpha_i k$). Assume that the $CreateTableValues$ subroutine is called using the set $E_l$ and that this routine is in the $k^{th}$ loop. If the 'if' condition is true, then value of $t_j$ is the same as $CreateTableValues$ and the values of $q'$ change as follows. Let $q'_{j_l} = k$ and $q'_{j_i} = q'_{j-\Sigma_{q=1}^{k} a_{f_{q_i}^i}}$ for $i = 1, ..., l-1$ where and $f_q^i$ is again the sorted order of the $E_i$ set according to its $a$ coefficients.

The main idea behind the $q'$ is to know how many elements from each set must have variables set to one in order for $\Sigma_{i=1}^{l} \Sigma_{j \in E_i} \alpha_i x_j = \beta$. That is, the $q'$ are generated

such that $\Sigma_{i=1}^{l} q'_{j_i} \alpha_i = \beta$. Thus, the dimension of the simultaneously up lifted inequality increases its dimension by at least the number of affinely independent points that meet these $q'$ restrictions for each set. Regardless, this dimension is guaranteed to increase by at least one. Formally,

**Theorem 3.1.2.** *If $\Sigma_{j=1}^{n} a_j x_j \leq b$ is a knapsack constraint, a finite $\alpha^*$ is taken from the DPMLSA and $\Sigma_{j \in N \setminus E_p} \alpha_j x_j \leq \beta$ defines a face of dimension $r$ over $PKP_{E_p}^{ch}$, then $\Sigma_{j \in E_p} \alpha^* x_j + \Sigma_{j \in N \setminus E_p} \alpha_j x_j \leq \beta$ defines a face of at least $r+1$ in $PKP_{E_p}^{ch}$.*

*Proof:* Assume $\Sigma_{j=1}^{n} a_j x_j \leq b$ is a knapsack constraint, a finite $\alpha^*$ is taken from the DPMLSA and $\Sigma_{j \in N \setminus E_p} \alpha_j x_j \leq \beta$ defines a face of dimension $r$ over $PKP_{E_p}^{ch}$. Since this is a face over $PKP_{E_p}^{ch}$, there exist $r+1$ affinely independent points that have $x_j = 0$ for all $j \in E_p$ and each point satisfies $\Sigma_{j \in N \setminus E_p} \alpha_j x_j = \beta$ . Therefore, it suffices to find a single feasible point, $x^*$ such that $\Sigma_{j \in N} \alpha_j x_j^* \leq \beta$ and there exists at least one $j \in E^P$ such that $x_j* = 1$.

Since $\alpha^* < \infty$, the 'if' condition *if* $t_j < t_{j - \Sigma_{q=1}^{k} a_{f_q^p}} + \alpha_i k$ had to be satisfied for some $j$. Thus, there exists a feasible point $x^* \in P$ such that $\Sigma_{j \in E_p} x_j^* \geq 1$ and $\Sigma_{j \in N} \alpha_j x_j^* = \beta$. This point has precisely $q'_{b_l}$ variables set to one where the variables correspond to the indices in $E_l$ with the smallest coefficients for $l = 1, ..., p$ and the result follows.

□

Besides generating a valid inequality, the DPMLSA also guarantees to increase the dimension of the face by at least one in the expanded polyhedron. Under some more restrictive conditions, as seen in the next theorem, a facet defining result is possible.

**Theorem 3.1.3.** *If $\Sigma_{j=1}^{n} a_j x_j \leq b$ is a knapsack constraint, a finite $\alpha^*$ is taken from the DPMLSA and $\Sigma_{j \in N \setminus E_p} \alpha_j x_j \leq \beta$ is a facet defining inequality over $PKP_{E_p}^{ch}$, then $\Sigma_{j \in E_p} \alpha^* x_j + \Sigma_{j \in N \setminus E_p} \alpha_j x_j \leq \beta$ is facet defining over $PKP^{ch}$ if the following conditions are met:*

*(i)* $1 \leq q'_{b_p} \leq |E_p| - 1$,

*(ii)* $\Sigma_{j=2}^{q'_{b_p}+1} a_{f_j^p} + \Sigma_{i=1}^{p-1} \Sigma_{j=1}^{q'_{b_i}} a_{f_j^i} \leq b$ *and*

*(iii)* $a_{f^p_{|E_p|}} + \Sigma_{j=1}^{q'_{b_p}-1} a_{f_j^p} + \Sigma_{i=1}^{p-1} \Sigma_{j=1}^{q'_{b_i}} a_{f_j^i} \leq b$ *as long as* $q'_{b_p} \leq |E_p| - 2$.

*Proof:* Assume $\Sigma_{j=1}^{n} a_j x_j \leq b$ is a knapsack constraint, a finite $\alpha^*$ is taken from the DPMLSA and (i), (ii) and (iii) are true. Furthermore, assume $\Sigma_{j \in N \setminus E_p} \alpha_j x_j \leq \beta$ defines a facet over $PKP_{E_p}^{ch}$. Thus there exist $|N \setminus E_p| + 1$ affinely independent points in $PKP$ that satisfy $\Sigma_{j \in E_p} \alpha^* x_j + \Sigma_{j \in N \setminus E_p} \alpha_j x_j = \beta$ and have $x_j = 0$ for all $j \in E_p$. So it suffices to find an additional $|E_p|$ affinely independent points in $PKP$ that meet this inequality at equality.

If $q'_{b_p} = 1$, then $a_{f^p_{|E_p|}} + \Sigma_{i=1}^{p-1} \Sigma_{j=1}^{q'_{b_i}} a_{f_j^i} \leq b$ and conditions (ii) and (iii) are met. Thus the point $e_{f^p_{|E_p|}} + \Sigma_{i=1}^{p-1} \Sigma_{j=1}^{q'_{b_i}} e_{f_j^i}$ is in $PKP$ where $e_j$ is the $j^{th}$ identity point. Due to the sorted order of $E_p$, $e_{f_{j'}^p} + \Sigma_{i=1}^{p-1} \Sigma_{j=1}^{q'_{b_i}} e_{f_j^i}$ is in $PKP$ for any $j' \in E_p$. Clearly, each of these $|E_p|$ points satisfy $\Sigma_{j \in E_p} \alpha^* x_j + \Sigma_{j \in N \setminus E_p} \alpha_j x_j = \beta$ and this case is shown.

If $q'_{b_p} = |E_p| - 1 \geq 2$, then $\Sigma_{j=2}^{q'_{b_p}+1} a_{f_j^p} + \Sigma_{i=1}^{p-1} \Sigma_{j=1}^{q'_{b_i}} a_{f_j^i} \leq b$ by (ii). Thus, $\Sigma_{j=1}^{q'_{b_p}+1} a_{f_j^p} - a_{f_{j'}^p} + \Sigma_{i=1}^{p-1} \Sigma_{j=1}^{q'_{b_i}} a_{f_j^i} \leq b$ for any $j' = 1, ..., |E_p|$ due to the sorted order of $F_p$. Thus, the points $\Sigma_{j=1}^{q'_{b_p}+1} e_{f_j^p} - e_{f_{j'}^p} + \Sigma_{i=1}^{p-1} \Sigma_{j=1}^{q'_{b_i}} e_{f_j^i}$ are in $PKP$ for any $j' = 1, ..., |E_p|$. Additionally, these points meet $\Sigma_{j \in E_p} \alpha^* x_j + \Sigma_{j \in N \setminus E_p} \alpha_j x_j \leq \beta$ at equality and are affinely independent

and the result follows.

For the final case, if $q'_{b_p} < |E_p| - 1$ and $q'_{b_p} \geq 2$, then $\Sigma_{j=2}^{q'_{b_p}+1} a_{f_j^p} + \Sigma_{i=1}^{p-1}\Sigma_{j=1}^{q'_{b_i}} a_{f_j^i} \leq b$ (ii). Thus, $\Sigma_{j=1}^{q'_{b_p}+1} a_{f_j^p} - a_{f_{j'}^p} + \Sigma_{i=1}^{p-1}\Sigma_{j=1}^{q'_{b_i}} a_{f_j^i} \leq b$ for any $j' = 1, ..., q'_{b_p} + 1$ due to the sorted order of $F_p$. Thus, the points $\Sigma_{j=1}^{q'_{b_p}+1} e_{f_j^p} - e_{f_{j'}^p} + \Sigma_{i=1}^{p-1}\Sigma_{j=1}^{q'_{b_i}} e_{f_j^i}$ are in $PKP$ for any $j' = 1, ..., q'_{b_p} + 1$. Additionally, these points meet $\Sigma_{j\in E_p}\alpha^* x_j + \Sigma_{j\in N\setminus E_p}\alpha_j x_j \leq \beta$ at equality.

To find the remaining $|E_p| - q'_{b_p}$ points observe that $a_{f_{|E_p|}^p} + \Sigma_{j=1}^{q'_{b_p}-1} a_{f_j^p} + \Sigma_{i=1}^{p-1}\Sigma_{j=1}^{q'_{b_i}} a_{f_j^i} \leq b$. Thus, $a_{f_{j'}^p} + \Sigma_{j=1}^{q'_{b_p}-1} a_{f_j^p} + \Sigma_{i=1}^{p-1}\Sigma_{j=1}^{q'_{b_i}} a_{f_j^i} \leq b$ for any $j' \geq |E_p| - q'_{b_p} + 1$. Consequently, the points $e_{f_{j'}^p} + \Sigma_{j=1}^{q'_{b_p}-1} e_{f_j^p} + \Sigma_{i=1}^{p-1}\Sigma_{j=1}^{q'_{b_i}} e_{f_j^i}$ are in $PKP$ for any $j' \geq |E_p| - q'_{b_p} + 1$ and meet $\Sigma_{j\in E_p}\alpha^* x_j + \Sigma_{j\in N\setminus E_p}\alpha_j x_j \leq \beta$ at equality.

These points are clearly affinely independent and the final case follows.

□

Therefore, it has been shown that the DPMLSA runs with $O(nb)$ effort, and generates the valid inequality $\Sigma_{i=1}^{p}\alpha_i\Sigma_{j\in E_i} x_j \leq \beta$. The DPMLSA also guarantees to increase the dimension of the face by at least one in the expanded polyhedron, and a facet defining result is possible.

To create an example problem where three sets can be simultaneously lifted, the IP from Example 2 is expanded by introducing more variables. The example can be found below.

**Example 3**

Now let $A = [37, 36, 36, 35, 34, 23, 23, 22, 22, 21, 21, 20, 20, 15, 15, 15, 14, 14, 14, 13, 13, 13, 12, 12, 12]$ and $b = 152$. Let the sets be $E_1 = \{1, 2, ..., 5\}$,$E_2 = \{6, 7, ..., 13\}$ and $E_3 = \{14, 15, ..., 25\}$. Observe that each of these sets is a cover. Here we detail the generation of one simultaneously lifted inequality. This inequality starts from the cover inequality generated from $E_1$ and simultaneously lifts $E_2$ and then $E_3$. Thus, the final inequality takes the form $\Sigma_{j=1}^{5} x_j + \alpha_2 \Sigma_{j \in E_2} x_j + \alpha_3 \Sigma_{j \in E_3} x_j \leq 4$.

The table from $E_1$ must be generated in order to lift $E_2$. Since the smallest $a$ value in $E_1$ is 34, the value of $\Sigma_{j=1}^{5} x_j$ is zero as long as the right hand side is less than or equal to 33. Thus, $t_j = 0$ $\forall j = 0$ to 33. Once the right hand side is between 34 and 68, at most one of the variables ($x_5 = 1$) corresponding to an index from $E_1$ can be set to one, so $t_j = 1$ for all $j = 34$ to 68 ($x_5 = 1$). Continuing this logic yields $t_j = 2$ for all $j = 69$ to 104, $t_j = 3$ for all $j = 105$ to 140, $t_j = 4$ for all $j = 141$ to 152. Observe that these ranges are the sums of the smallest coefficients in $E_1$.

To determine $\alpha_2$ let $F_2$ be $E_2$ sorted in ascending order. Thus, the $a$ values of the $F_2$ indices are $(20, 20, 21, 21, 22, 22, 23, 23)$. Set $\alpha_2 = \infty$ and $k = 1$. Now $a_{f_1} = 20$ and $b - a_{f_1} = 132$, so $\frac{\beta - t_{132}}{1} = \frac{4-3}{1} = 1$, which is less than $\infty$. So $\alpha_2$ is changed to 1. Similarly, $a_{f_1^2} + a_{f_2^2} = 40$ and $b - a_{f_1^2} - a_{f_2^2} = 112$, so $\frac{\beta - t_{112}}{2} = \frac{4-3}{2} = \frac{1}{2}$ and $\alpha_2$ is set to $\frac{1}{2}$. Continuing this pattern for the smallest three through the smallest 7 indices of $E_3$ does not change the value of $\alpha_2^*$ because $\frac{\beta - t_{91}}{3} = \frac{4-2}{3} = \frac{2}{3}$, $\frac{\beta - t_{70}}{4} = \frac{4-2}{4} = \frac{1}{2}$, $\frac{\beta - t_{48}}{5} = \frac{4-1}{5} = \frac{3}{5}$, $\frac{\beta - t_{26}}{6} = \frac{4-0}{6} = \frac{4}{6}$, and $\frac{\beta - t_3}{7} = \frac{4-0}{7} = \frac{4}{7}$, which are all larger than $\frac{1}{2}$. So simultaneously lifting $E_2$ into the cover inequality generated by $E_1$ results in

$x_1 + x_2 + x_3 + x_4 + x_5 + \frac{1}{2}x_6 + \frac{1}{2}x_7 + \frac{1}{2}x_8 + \frac{1}{2}x_9 + \frac{1}{2}x_{10} + \frac{1}{2}x_{11} + \frac{1}{2}x_{12} + \frac{1}{2}x_{13} \leq 4$.

In order to simultaneously lift $E_3$, $T$ is updated with $E_2$. Now that $\alpha_2^* = \frac{1}{2}$, $T$ can be updated by adding the $\alpha_2$ value to the table. This is done by evaluating the statement if $t_j + \alpha_2 \geq t_{j+f_1^2}$, then $t_{j+f_1^2} = t_j + \alpha_2$. First, consider updating the table with $a_{f_1^2}$. Compare $t_0 + \alpha_2^*$ to $t_{20}$, since $(t_0 + \alpha_2^* = \frac{1}{2}) > (t_{20} = 0)$, $t_{20} = \frac{1}{2}$. Incrementing $j$ results in $t_{20}$ to $t_{33}$ being updated and set equal to $\frac{1}{2}$, $t_{54}$ to $t_{68}$ updated to $1\frac{1}{2}$, $t_{89}$ to $t_{104}$ updated to $2\frac{1}{2}$, and $t_{125}$ to $t_{140}$ updated to $3\frac{1}{2}$ and all the remaining values of $T$ remain the same.

To continue updating $T$, the sum of the smallest two coefficients corresponding to indices from $E_2$ are considered. Observe that the sum of these two coefficients, $a_{f_1^2} + a_{f_2^2}$, is 40, which is larger than any element in $E_1$ and indicates that no updates should be generated from this pair. The first check would be to compare $(t_{0+f_1^2+f_2^2} = t_{40} = 1)$ to $(t_0 + 2\alpha_2 = 1)$ and observe that no update occurs.

Continuing this logic, no additional updates to $T$ occur from elements in $E_2$, and $T$ becomes $t_j = 0 \ \forall j = 0$ to 19, $t_j = 0.5 \ \forall \ j = 20$ to 33, $t_j = 1 \ \forall \ j = 34$ to 53, $t_j = 1.5 \ \forall \ j = 54$ to 68, $t_j = 2 \ \forall \ j = 69$ to 88, $t_j = 2.5 \ \forall \ j = 89$ to 104, $t_j = 3 \ \forall \ j = 105$ to 124, $t_j = 3.5 \ \forall \ j = 125$ to 140, and $t_j = 4 \ \forall \ j = 141$ to 152.

To determine $\alpha_3$ let $F_3$ be $E_3$ sorted in ascending order. Thus, the $a$ values of the $F_3$ indices are $(12, 12, 12, 13, 13, 13, 14, 14, 14, 15, 15, 15)$. Set $\alpha_3 = \infty$ and $k = 1$. Now $a_{f_1^3} = 12$ and $b - a_{f_1^3} = 140$, so $\frac{\beta - t_{140}}{1} = \frac{4-3.5}{1} = \frac{1}{2}$ and $\alpha_3 = \frac{1}{2}$. Again, $a_{f_1^3} + a_{f_2^3} = 24$ and $b - a_{f_1^3} - a_{f_2^3} = 128$ so $\frac{\beta - t_{128}}{2} = \frac{4-3.5}{2} = \frac{1}{4}$ and $\alpha_3 = \frac{1}{4}$. This process is continued to find the remaining changes; they are $\frac{\beta - t_{116}}{3} = \frac{4-3}{3} = \frac{1}{3}$, $\frac{\beta - t_{103}}{4} = \frac{4-2.5}{4} = \frac{3}{8}$, $\frac{\beta - t_{90}}{5} =$

$\frac{4-2.5}{5} = \frac{3}{10}$, $\frac{\beta - t_{77}}{6} = \frac{4-2}{6} = \frac{1}{3}$, $\frac{\beta - t_{63}}{7} = \frac{4-1.5}{7} = \frac{5}{14}$, $\frac{\beta - t_{49}}{8} = \frac{4-1}{8} = \frac{3}{8}$, $\frac{\beta - t_{35}}{9} = \frac{4-1}{9} = \frac{1}{3}$, $\frac{\beta - t_{20}}{10} = \frac{4-.5}{10} = \frac{3.5}{10}$, and $\frac{\beta - t_5}{11} = \frac{4-0}{11} = \frac{4}{11}$. The minimum has $\alpha_3^* = \frac{1}{4}$.

So the final simultaneously lifted inequality is $\Sigma_{j \in E_1} x_j + \alpha_2^* \Sigma_{j \in E_2} x_j + \alpha_3^* \Sigma_{j \in E_3} x_j \leq 4$, which is $\Sigma_{j=1}^{5} x_j + \frac{1}{2} \Sigma_{j=6}^{13} x_j + \frac{1}{4} \Sigma_{j=14}^{25} x_j \leq 4$. The following discussion shows that this inequality satisfies Theorem 3.1.3 and is therefore facet defining for this polytope.

Example 3 starts with the set $E_1$ which is a minimal cover of $PKP^{ch}$. As a minimal cover, the dimension of the cover inequality is 4 in $PKP^{ch}_{E_2 \cup E_3}$. The first five columns of Figure 3.1 below give the five affinely independent points necessary to prove this dimension.

By simultaneously lifting $E_2$ into the cover inequality, this dimension of the restricted space is expanded to 12. The $\alpha_2^* = \frac{1}{2}$ occurs by combining the three smallest elements of $E_1$ with two elements of $E_2$, thus $q'_{b_1} = 3$ and $q'_{b_2} = 2$. Consequently, having three variables in $E_1$ and two variables in $E_2$ set to one results in $\Sigma_{j=1}^{5} x_j + \frac{1}{2} \Sigma_{j=6}^{13} x_j \leq 4$ being met at equality.

The next eight columns of affinely independent points show that the dimension is increased to 12 by $E_2$ being lifted. From Theorem 3.1.3, $(i)$ is clearly met since $q'_{b_2} = 2 < 7$. Since $q'_{b_1}$ represents the number of elements of $E_1$ to be set to 1 and $q'_{b_2}$ represents the same for $E_2$, these points meet the inequality $(ii)$ at equality. The sixth point, $[0,0,1,1,1,1,0,0,0,0,\ 0,0,1,0,0,0,0,0,0,0,0,0,0,0,0]$ is feasible, because the sum of the coefficients of the elements is $36 + 35 + 34 + 23 + 20 = 148$ which is less than 152, which satisfies condition $(iii)$. Clearly, if this point is feasible, then replacing the element

$x_6$ with any other element of $E_2$ is feasible. Therefore, the seventh through tenth points are feasible by the sorted order.

Similarly, the eleventh point $[0,0,1,1,1,0,0,0,0,\ 0,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0]$ is feasible. The sum of the coefficients for this point is $36+35+34+21+20=146$ which is less than 152, thus condition $(ii)$ is met. Since this point is feasible, the twelfth and thirteenth points are also feasible by the sorted order of $a_j$.

The set $E_3$ is then simultaneously lifted into the inequality, and the lowest $\alpha$ value occurs with two elements of $E_3$, one element of $E_2$, and two elements of the original cover inequality, $E_1$. Thus, $q'_{b_1}=3$, $q'_{b_2}=1$, and $q'_{b_3}=2$.

This combination of elements clearly satisfies condition $(i)$ of Theorem 3.1.3. Since the points listed in the columns below meet $(ii)$ at equality, condition $(ii)$ is met. Points fourteen through twenty two are feasible. This can be shown by proving point fourteen, $[0,0,1,1,1,0,0,0,0,0,0,1,1,0,0,0,0,0,0,0,0,0,0,0,1]$, is feasible, because the sum of the coefficients $36+35+34+20+15+12=152$, which satisfies the constraint. Points twenty three, twenty four, and twenty five are feasible; this is shown by proving the largest is feasible. Point twenty three is feasible because the sum of the coefficients $36+35+34+20+12+12=149$ which is less than 152.

The twenty five affinely independent points in Figure 3.1 below prove that the inequality generated by the DPMLSA, $\Sigma_{j=1}^{5}x_j+\frac{1}{2}\Sigma_{j=6}^{13}x_j+\frac{1}{4}\Sigma_{j=14}^{25}x_j\leq 4$, is valid and facet defining for $PKP^{ch}$.

| | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

Figure 3.1: Affinely Independent Points for Example 3

The DPMLSA provides a fast method for finding valid simultaneously up lifted inequalities, and the inequalities have the potential to be facet defining. The DPMLSA runs in pseudo-polynomial, $O(nb)$, effort, which is fast in most instances. In some instances, $b$ may be exponentially large; for these instances a polynomial time algorithm would be better suited for generating the simultaneously up lifted inequality, which is the topic of the next section.

## 3.2 Three Set Simultaneous Lifting Algorithm

In some instances, $b$ can be exponentially large, which forces the $O(nb)$ effort of the DPMLSA to be computationally intractable. For these instances, an algorithm that runs independently of $b$ is desirable. The effort required for the TSSLA depends on the number of sets simultaneously up lifted; for three sets, the TSSLA requires $O(n^2)$.

The input to the Three Set Simultaneous Lifting Algorithm (TSSLA) is a knapsack constraint $\Sigma_{j\in N} a_j x_j \leq b$, a lifting set $E_3 \subset N$ and a valid inequality of the form $\alpha_1 \Sigma_{j\in E_1} x_j + \alpha_2 \Sigma_{j\in E_2} x_j \leq \beta$ over $PKP^{ch}_{E_3}$ where $E_1, E_2$ and $E_3$ are all disjoint.

**Three Set Simultaneous Lifting Algorithm**

**Initialization**

Let $F_k = \{f^k_1, ..., f^k_{|E_k|}\}$ where $F_k$ is $E_k$ sorted in ascending order based upon the corresponding $a$ values of the knapsack constraint for $k = 1, ..., 3$.

$\alpha_3 := \infty$.

**Main Step**

*For* $r = 1$ *to* $|E_3|$

$p := 0$, $q := 0$ and $sum := \Sigma^r_{k=1} a_{f^3_k}$.

*While* $sum \leq b$ *and* $p \leq |E_1|$

$p := p + 1$

$sum := sum + a_{f^1_p}$

$flag := feasible$

*End While*

$sum := sum - a_{f_p^1}$ and $p := p - 1$

*While* $p \geq 0$ *and* $q \leq |E_2|$.

*if* $sum \leq b$, *then*

*if* $(\beta - p\alpha_1 - q\alpha_2)/r < \alpha_3$, *then* $\alpha_3 = \frac{(\beta - p\alpha_1 - q\alpha_2)}{r}$.

$q := q + 1$ and $sum := sum + a_{f_q^2}$.

*else*

$sum := sum - a_{f_p^1}$ and $p := p - 1$.

*End While*

*End For*

**Output**

Report $\alpha_3$ as the simultaneous lifting coefficient.

To determine the running time of the TSSLA the algorithm will be evaluated step by step. First, consider the Initialization Step. The sorting of the three $E_p$ sets requires $O(|E|log(|E|))$ where $|E| = max\{|E_i| : i = 1, 2, 3\}$. So, overall, the Initialization requires $O(|E|(log(|E|)))$ effort.

Next, the Main Step is evaluated. The summation of elements in $E_1$ requires $O(|E_1|)$,

and the next while loop requires $O((|E_1|+|E_2|)(|E_3|))$. Finally, the termination requires $O(1)$. Using amortized analysis, the TSSLA runs in $O((|E_1|+|E_2|)(|E_3|))$ effort which is bounded by $O(n^2)$ effort.

Implementing the TSSLA results in simultaneously up lifted inequality that is a valid inequality of $PKP^{ch}$. An additional benefit of this inequality is that its dimension in $PKP^{ch}$ is larger than the dimension of the original inequality in $PKP^{ch}_{E_3}$. Formally,

**Theorem 3.2.1.** *Let $\Sigma_{j=1}^{n} a_j x_j \leq b$ be a knapsack constraint and $E_1$, $E_2$ and $E_3$ be nonempty, disjoint subsets of $N$. If $\alpha_1 \Sigma_{j\in E_1} x_j + \alpha_2 \Sigma_{j\in E_2} x_j \leq \beta$ is a valid inequality over $PKP^{ch}_{E_3}$, then the inequality $\alpha_1 \Sigma_{j\in E_1} x_j + \alpha_2 \Sigma_{j\in E_2} x_j + \alpha_3 \Sigma_{j\in E_3} x_j \leq \beta$ is valid for $PKP^{ch}$ as long as $\alpha_3$ is taken from the TSSLA. Furthermore, if $\alpha_1 \Sigma_{j\in E_1} x_j + \alpha_2 \Sigma_{j\in E_2} x_j + \alpha_3 \Sigma_{j\in E_3} x_j \leq \beta$ defines a face of dimension $r$ over $PKP^{ch}_{E_3}$, then $\alpha_1 \Sigma_{j\in E_1} x_j + \alpha_2 \Sigma_{j\in E_2} x_j + \alpha_3 \Sigma_{j\in E_3} x_j \leq \beta$ defines a face of dimension at least $r+1$ over $PKP^{ch}$.*

**Proof:** Assume $\Sigma_{j=1}^{n} a_j x_j \leq b$ is a knapsack constraint, $E_1$, $E_2$ and $E_3$ are nonempty, disjoint subsets of $N$, $\alpha_1 \Sigma_{j\in E_1} x_j + \alpha_2 \Sigma_{j\in E_2} x_j \leq \beta$ is a valid inequality over $PKP^{ch}_{E_3}$, and the TSSLA returns $\alpha_3$ as the lifting coefficient. For contradiction, assume that $\alpha_1 \Sigma_{j\in E_1} x_j + \alpha_2 \Sigma_{j\in E_2} x_j + \alpha_3 \Sigma_{j\in E_3} x_j \leq \beta$ is not valid for $PKP^{ch}$. Thus, there exists an $x' \in PKP$ such that $\alpha_1 \Sigma_{j\in E_1} x'_j + \alpha_2 \Sigma_{j\in E_2} x'_j + \alpha_3 \Sigma_{j\in E_3} x'_j > \beta$.

Let $q'_i = |\{x'_j = 1 : j \in E_i\}|$ for $i = 1, ..., 3$. Clearly, $q'_3 > 0$ since $\alpha_1 \Sigma_{j\in E_1} x_j + \alpha_2 \Sigma_{j\in E_2} x_j \leq \beta$ is a valid inequality over $PKP^{ch}_{E_3}$. In the TSSLA loop when $r = q'_3$, there is an iteration that has $p = q'_1$ and $q = q'_2$. Since this point is feasible, $\alpha_3 \leq \frac{(\beta - p\alpha_1 - q\alpha_2)}{r} = \frac{(\beta - q'_1\alpha_1 - q'_2\alpha_2)}{q'_3}$. However, this implies that $x'$ does not violate $\alpha_1 \Sigma_{j\in E_1} x_j + \alpha_2 \Sigma_{j\in E_2} x_j +$

$\alpha_3\Sigma_{j\in E_3}x_j \leq \beta$, a contradiction.

To show that the dimension of the face defined by $\alpha_1\Sigma_{j\in E_1}x_j+\alpha_2\Sigma_{j\in E_2}x_j+\alpha_3\Sigma_{j\in E_3}x_j \leq \beta$ increases by at least one, observe that the point with the variables set to one for the smallest $q_i'$ elements in each $E_i$ for $i = 1, ..., 3$ meets this inequality at equality. Since $q_3' > 0$, the dimension must increase by at least one since the space is expanding from $PKP_{E_3}^{ch}$ to $PKP^{ch}$.

□

Implementing the TSSLA provides the same exact simultaneously up lifted inequality as the DPMLSA. This is due to the fact that the same three sets, $E_1, E_2, E_3$, are lifted in the same order. Therefore, the inequality generated by the TSSLA is facet defining by Theorem 3.1.3 for the same reasons as the inequality generated by the DPMLSA. The following example shows how to implement the TSSLA.

**Example 4**

The problem from Example 3 is used for this example of the Three Set Simultaneous Lifting Algorithm.

Recall that $A = [37, 36, 36, 35, 34, 23, 23, 22, 22, 21, 21, 20, 20, 15, 15, 15, 14, 14, 14, 13, 13, 13, 12, 12, 12]$ and $b = 152$. The sets are $E_1 = \{1, 2, ..., 5\}$, $E_2 = \{6, 7, ..., 13\}$, and $E_3 = \{14, 15, ..., 25\}$, and each set defines a cover. The starting equation is $\Sigma_{j=1}^{5}x_j + \frac{1}{2}\Sigma_{j=6}^{13}x_j \leq 4$, which is equivalent to $x_1 + x_2 + x_3 + x_4 + x_5 + \frac{1}{2}x_6 + \frac{1}{2}x_7 + \frac{1}{2}x_8 + \frac{1}{2}x_9 + \frac{1}{2}x_{10} + \frac{1}{2}x_{11} + \frac{1}{2}x_{12} + \frac{1}{2}x_{13} \leq 4$. Notice that this equation could have been found by lifting in $E_2$ in linear time with Hooker's or Sharma's method.

The first step of the TSSLA would be to add the elements of $E_1$ to $a_{f_1^3}$ until they are greater than the right hand side of the inequality, $b = 152$. The sum starts off as $a_{f_1^3} = 12$ and elements of $E_1$ are added to the sum until $sum = 153$. Then, the last element added is removed, decreasing the sum to 117; this ensures that the largest number of variables set to 1 in $E_1$, in this case 4, are used while keeping the sum less than $b$.

Next, an equation is evaluated to see if the value is less than $\alpha_3$; if so, $\alpha_3$ is modified to reflect the new value. For this example, $\frac{4-3(1)-0(\frac{1}{2})}{1} = 1$; therefore, $\alpha_3 = 1$. Now, an element of $E_2$ is added to make $sum = 137$, $\frac{4-3(1)-1(\frac{1}{2})}{1} = \frac{1}{2}$, and $\alpha_3 = \frac{1}{2}$. The next coefficient of an element of $E_2$, 20, is added to make $sum = 157$; since this is greater than 152, the else statement comes into effect and an element of $E_1$ is removed. Now, $sum = 121$, $\frac{4-2(1)-2(\frac{1}{2})}{1} = 1$, and $\alpha_3$ is not updated. This process is repeated for all critical combinations of $E_1$ and $E_2$ with the first element of $E_3$. The process continues as follows: $sum = 142$ and $\frac{4-2(1)-3(\frac{1}{2})}{1} = \frac{1}{2}$, $sum = 163$ so an element of $E_1$ is removed, $sum = 128$ and $\frac{4-1(1)-4(\frac{1}{2})}{1} = 1$, $sum = 150$ and $\frac{4-1(1)-5(\frac{1}{2})}{1} = \frac{1}{2}$, $sum = 172$ so the last element of $E_1$ is removed, $sum = 138$ and $\frac{4-0(1)-6(\frac{1}{2})}{1} = 1$, and finally $sum = 161$. One complete loop of the Main Step is completed, with $\alpha_3 = \frac{1}{2}$.

This cycle is completed with an additional element of $E_3$ each time until all elements of $E_3$ have been combined and evaluated. With two elements from $E_3$ and three elements of $E_1$, $sum = 129$ to begin with. The process continues as follows: $sum = 129$ and $\frac{4-3(1)-0(\frac{1}{2})}{2} = \frac{1}{2}$, $sum = 149$, $\frac{4-3(1)-1(\frac{1}{2})}{2} = \frac{1}{4}$, and $\alpha_3 = \frac{1}{4}$, $sum = 169$ so an element

of $E_1$ is removed, $sum = 133$ and $\frac{4-2(1)-2(\frac{1}{2})}{2} = \frac{1}{2}$, $sum = 154$ an element of $E_1$ is removed, $sum = 119$ and $\frac{4-1(1)-3(\frac{1}{2})}{2} = \frac{3}{4}$, $sum = 140$ and $\frac{4-1(1)-4(\frac{1}{2})}{2} = \frac{1}{2}$, $sum = 162$ and the last element of $E_1$ is removed, $sum = 128$ and $\frac{4-0(1)-5(\frac{1}{2})}{2} = \frac{3}{4}$, $sum = 150$ and $\frac{4-0(1)-6(\frac{1}{2})}{2} = \frac{1}{2}$, and $sum = 173$. This loop of the Main Step resulted in $\alpha_3 = \frac{1}{4}$.

The other ten cycles of the Main Step would be conducted in the same way and the final result would remain $\alpha_3 = \frac{1}{4}$. This would provide the final valid inequality of $\alpha_1 \Sigma_{j \in E_1} x_j + \alpha_2 \Sigma_{j \in E_2} x_j + \alpha_3 \Sigma_{j \in E_3} x_j \leq 4$, which is $\Sigma_{j=1}^{5} x_j + \frac{1}{2} \Sigma_{j=6}^{13} x_j + \frac{1}{4} \Sigma_{j=14}^{25} x_j \leq 4$. As shown in Example 3, for the DPMLSA, the final valid inequality is facet defining over $PKP^{ch}$.

The TSSLA produces valid simultaneously up lifted inequalities. The inequality generated by the TSSLA is facet defining if the conditions of Theorem 3.1.3 are met. The TSSLA generates inequalities in polynomial, $O(n^2)$, effort. This is fast in all instances and is preferable to the DPMLSA if $b$ is large.

### 3.2.1 Extensions of the TSSLA

The algorithm, TSSLA, can be expanded to include an arbitrary number of sets. In order to extend this algorithm to multiple sets, another loop statement would need to be added to accommodate the additional set $E_4$. The effort required would increase by the order of $n$ for every additional set. Therefore, in general, the effort required to implement this algorithm is at most $O(n^{p-1})$ for $p$ sets.

Observing that these $p$ sets are bounded by $n$, the running time is at most $O(n^{n-1})$.

However, recognizing that these $p$ sets partition $n$ elements, this running time can be reduced further. The maximum running time occurs by solving $max\Pi_{i=1}^{p}|E_i|$ subject to $\cup_{i=1}^{p}E_i = N$ and $E_i \cap E_j = \emptyset$ for all $i \neq j$. The maximum occurs when each $E_i$ contains $\sqrt{n}$ elements. Thus, regardless of the number of partitions, the TSSLA is bounded by $O(\sqrt{n}^{\sqrt{n}})$ effort.

## 3.3 Sequential Simultaneous Lifting

One of the most significant results of this research is the introduction of a new type of lifting, sequential simultaneous lifting. For all the examples in this thesis, the sets were evaluated in the same order: $E_1$, then $E_2$, and then $E_3$. This was kept consistent because the inequality generated by these algorithms is sequence dependent. If the three sets were lifted in a different order each time, up to $3! = 6$ distinct inequalities could be reported by the DPMLSA or the TSSLA.

Theoretically, a problem with $p$ sets could provide $p!$ distinct exact simultaneously up lifted inequalities. This would require the lifting order of the sets be changed for every evaluation. For example, the six lifting combinations for a problem with three sets would be $(1^{st}, 2^{nd}, 3^{rd}), (1^{st}, 3^{rd}, 2^{nd}), (3^{rd}, 1^{st}, 2^{nd}),$ $(2^{nd}, 1^{st}, 3^{rd}), (3^{rd}, 2^{nd}, 1^{st}),$ and $(2^{nd}, 3^{rd}, 1^{st})$.

The 25 variable example used throughout this thesis would produce six distinct inequalities, if the right hand side is changed to $b = 149$. The six inequalities are

(**1**) $\Sigma_{j=1}^{5} x_j + \frac{1}{2}\Sigma_{j=6}^{13} x_j + \frac{1}{4}\Sigma_{j=14}^{25} x_j \leq 4$,

(**2**) $\Sigma_{j=1}^{5} x_j + \frac{1}{3}\Sigma_{j=6}^{13} x_j + \frac{1}{3}\Sigma_{j=14}^{25} x_j \leq 4$,

(**3**) $\frac{7}{5}\Sigma_{j=1}^{5} x_j + \Sigma_{j=6}^{13} x_j + \frac{3}{5}\Sigma_{j=14}^{25} x_j \leq 7$,

(**4**) $\frac{5}{3}\Sigma_{j=1}^{5} x_j + \Sigma_{j=6}^{13} x_j + \frac{1}{2}\Sigma_{j=14}^{25} x_j \leq 7$,

(**5**) $2\Sigma_{j=1}^{5} x_j + \frac{3}{2}\Sigma_{j=6}^{13} x_j + \Sigma_{j=14}^{25} x_j \leq 11$, and

(**6**) $\frac{5}{2}\Sigma_{j=1}^{5} x_j + \frac{5}{4}\Sigma_{j=6}^{13} x_j + \Sigma_{j=14}^{25} x_j \leq 11$.

Given that all the inequalities produced by these methods are valid, the potential for facet defining inequalities exist. The simultaneously up lifted inequalities would need to be evaluated with Theorem 3.1.3. Thus, this thesis introduced sequence dependent simultaneous up lifting.

In Examples 3 and 4, the initial set, $E_1$, is a minimal cover. This set is chosen because, as a minimal cover, it is facet defining over the restricted space $PKP^{ch}_{E_2 \cup E_3}$. After lifting sets $E_2$ and $E_3$, the exact simultaneously up lifted inequality would be facet defining over $PKP^{ch}$. However, it is not necessary for the DPMLSA or the TSSLA to start with a minimal cover, and it is possible to still achieve facet defining results.

For example, consider the constraint from Examples 3 and 4. If $x_1$ is removed, $A = [36, 36, 35, 34, 23, 23, 22, 22, 21, 21, 20, 20, 15, 15, 15, 14, 14, 14, 13, 13, 13, 12, 12, 12]$ for $x_2$ to $x_{25}$ and let $b$ be increased by 1, $b = 153$. Therefore, $E_1 = \{2, ..., 5\}$, $E_2 = \{6, 7, ..., 13\}$, and $E_3 = \{14, 15, ..., 25\}$. Clearly, the starting equation $\Sigma_{j=2}^{5} x_j \leq 4$ has dimension 0 in $P^{ch}_{E_2 \cup E_3}$, since the only point that meets it at equality is $[1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,$

$0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$.

Following either the DPMLSA or the TSSLA results in the same $\alpha$ values and the final valid inequality is $\Sigma_{j=2}^{5} x_j + \frac{1}{2}\Sigma_{j=6}^{13} x_j + \frac{1}{4}\Sigma_{j=14}^{25} x_j \leq 4$. The set of affinely independent points necessary to show this inequality is facet defining are

$$
\begin{array}{cccccccccccccccccccccccc}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1
\end{array}
.
$$

This example provides a significant result of exact simultaneous up lifting. It shows the ability of the DPMLSA and the TSSLA to increase the dimension of the starting inequality, dimension of 0, to facet defining over $PKP^{ch}$. This improvement is significant because the dimension increases by 23 by only simultaneously lifting 20 variables. Thus,

simultaneous lifting may increase the dimension of the polyhedron beyond the number of lifted variables.

# Chapter 4

# Computational Results

The most important result of this research is the ability to exact simultaneously up lift multiple sets in pseudo-polynomial and polynomial time. The purpose of this section is to provide computational results to support the findings of this research. The computational results show that valid inequalities generated by the DPMLSA are easy to find and can decrease the time required to solve an IP.

The computational study was conducted on a Pentium 4 3.40 GHz processor with 1.0 GB of RAM and times are reported in seconds. The study contained problems with 100 to 750 variables. All integer programs are solved using CPLEX at default settings.

The basic idea of the problems is to follow the spirit of the theoretical results [17, 45] to create difficult knapsack instances. These theoretical results have a $b$ value that is exponential in the size of the instance and since the DPMLSA's computational time relies on the order of $b$, sufficient memory is not available on KSU's computers to implement

this algorithm. Thus, the key ideas of these theoretical results are used to generate problems that CPLEX cannot trivially solve nor does it take CPLEX days to solve.

Various classes of instances were attempted before a suitable class of problems was discovered. This class of problems is a randomly generated multiple knapsack problem with two constraints. A multiple knapsack problem (MKP) is a KP with additional constraints. These constraints may represent the volume or budget requirements that further restrict the camper's choice. Thus, the IP formulation for this computational study follows the form maximize $\Sigma_{j=1}^{n} c_j x_j$ subject to $\Sigma_{j=1}^{n} a_{1j} x_j \leq b_1$, $\Sigma_{j=1}^{n} a_{2j} x_j \leq b_2$, and $x_j \in \{0, 1\}$ $\forall j \in \{1, ..., n\}$.

The constraint coefficients, $a_{ij}$, are random integers taken from a uniform distribution between 0 and $1,000$ for all $i \in \{1, 2\}$ and $j \in \{1, ..., n\}$. Each objective function coefficient, $c_j$, is calculated by adding the column coefficients and a uniform random integer between 0 and 5, $c_j = a_{1j} + a_{2j} + u$ where $u$ is a uniformly distributed integer between 0 and 5. The right hand side of each constraint is one tenth of the sum of the row's coefficients rounded down to the nearest integer, $\lfloor \Sigma_{j=1}^{n} \frac{a_{1j}}{10} \rfloor = b_1$ and $\lfloor \Sigma_{j=1}^{n} \frac{a_{2j}}{10} \rfloor = b_2$.

The computational study was conducted using $n$ variables, with $n$ equal to 100, 250, 500, and 750. To avoid anomalies with random instances, 30 instances of each size problem were generated and the averages are reported.

The DPMLSA was coded in C to generate an exact simultaneously lifted inequality for each of the two constraints. No effort was made to find good sets that eliminated the

root nodes of the linear relaxation. Rather, a minimal cover was obtained by taking a minimal cover consisting of the smallest $a$ coefficients for that constraint that are larger than 450. With this cover Balas's result [7] partitioned the variables into two classes, those that have fixed integer coefficients and those that are not fixed. Those with fixed integer coefficients are assigned to the correct value. The variables that are not fixed are used to form sets for the simultaneous up lifting as follows.

If the entire set of not fixed variables for a particular range is used for the DPMLSA, then the $\alpha$ value tended to be low. By dividing this set into several subsets, various $\alpha$ values were obtained. For instance, in one instance, lifting all of the set resulted in the inequality $2\ \Sigma_{j=1}^{2} x_j + 1\ \Sigma_{j=3}^{30} x_j + 0\ \sum_{j=31}^{50} x_j + \leq 5$. Once this was partitioned, the inequality became $2\ \Sigma_{j=1}^{2} x_j + 1.5\ \Sigma_{j=3}^{10} x_j + 1\ \Sigma_{j=11}^{30} x_j + \frac{1}{2}\ \Sigma_{j=31}^{36} x_j + \frac{1}{3}\ \Sigma_{j=37}^{42} x_j + 0\ \sum_{j=43}^{50} x_j + \leq 5$, which is clearly dominant.

After the $\alpha$ values are found, the two simultaneously up lifted inequalities, one for each constraint, are added to the original KP and solved using CPLEX 10.0 [66] at its default settings. For comparison, the program is run a second time without adding the two simultaneously lifted inequalities. Table 4.1 gives the average preprocessing time, the average solution time, and the percent improvement between these two runs.

To help validate that the computational study implemented the the DPMLSA correctly, the objective function value from CPLEX and the DPMLSA with CPLEX were compared. For these 120 instances the objective values were identical. This helps to confirm that the code successfully implemented the DPMLSA.

| | DPMLSA with CPLEX 10 at default | | | CPLEX 10 at default | | Percent Improvement |
|---|---|---|---|---|---|---|
| | Preprocessing Time | Average Time | Standard Deviation | Average Time | Standard Deviation | Average Time |
| n = 100 | 0.00 | 13.36 | 8.86 | 18.30 | 13.45 | 26.99% |
| n = 250 | 0.02 | 29.28 | 24.46 | 34.05 | 21.55 | 13.99% |
| n = 500 | 0.07 | 98.90 | 71.03 | 121.02 | 153.02 | 18.28% |
| n = 750 | 0.18 | 206.73 | 209.87 | 253.27 | 267.27 | 18.38% |
| Average | 0.069 | 87.07 | 134.11 | 106.67 | 179.0 | 18.4% |

Table 4.1: Comparison of Solution Time with the DPMLSA to without

Very little of the total running time can be attributed to the DPMLSA. The number of variables has a direct impact on the right hand side, $b$, which increases the time the DPMLSA takes to generate valid inequalities. The average preprocessing time for 100 variables was 0.00632 seconds; 250 variables, 0.0219 seconds; 500 variables, 0.0738 seconds; and 750 variables took 0.1779 seconds. As expected, an increasing trend is apparent with the increase of the number of variables and the right hand side, $b$. However, the DPMLSA is still extremely fast.

Overall, adding the two inequalities from the DPMLSA provided a significant improvement over CPLEX alone. This computational study shows that the DPMLSA is indeed fast, and the solution time to random knapsack instances decreased by an overall average of 18.4%. Another benefit of adding the constraints from the DPMLSA is that the standard deviation on the time required to solve each problem decreases, as expected.

A decrease of 18.4% is quite impressive given the random nature of the problems and the lack of effort to find a quality cover and useful sets. We attribute this success to

the fact that nearly all coefficients are non zero, simultaneously lifted inequalities tend to dominate the average of sequentially lifted inequalities, and these inequalities tend to be "further away" [46] from the axes than sequentially lifted cover inequalities.

At this point one may question why a computational study of the TSSLA was not performed. For starters, the TSSLA would generate the same inequality and so only the preprocessing time would change. Since this time is so small, it seems unnecessary to try and reduce this preprocessing time. Second, the DPMLSA simultaneously lifted 9 sets (3 for each set of ranges within Balas' result). Thus, the TSSLA would become a $O(n^8)$ algorithm and it is anticipated to require more preprocessing time than the DPMLSA. Thus, a computational study was not performed on the TSSLA.

# Chapter 5

# Conclusion and Future Research

The goal of this thesis was to develop fast methods to exact simultaneously up lift a third set into a cover inequality for the knapsack polytope. The algorithms presented in this thesis achieve this goal and surpass it. Additionally, this thesis presents a new type of lifting called sequential simultaneous lifting.

The Dynamic Programming Multiple Lifting Set Algorithm is solvable in $O(nb + nlog(n))$ effort. It could be argued that in most instances $b \geq log(n)$, and therefore, $O(nb)$ effort is required. The DPMLSA can be used for an arbitrary number of sets, but is primarily restricted by the size of $b$.

The Three Set Simultaneously Lifting Algorithm is solvable in $O(n^2)$ effort. This algorithm could be logically extended, with an additional loop statement for each additional set. This extension would require an order of $n$ for every additional set.

The DPMLSA and the TSSLA will generate the same inequality, if identical sets

are lifted in the same order. Neither algorithm requires the starting set to be a cover, and when the conditions of Theorem 3.1.3 are met, sequential simultaneously lifted inequalities generated by the DPMLSA and the TSSLA are facet defining.

## 5.1 Future Research

From a theoretical view point, simultaneous up lifting over binary knapsack polyhedra is now completely understood. Due to this research, fast algorithms and facet defining results now exist, and no further research into sequential simultaneous up lifting over binary knapsack polyhedra is necessary.

However, there still exists ample work for future research in computational results and determining what sets should be simultaneously lifted. Research could also be conducted on extending these concepts to down lifting and non binary problems. Future research could also be conducted to extended these concepts to mixed integer programs.

Opportunities for research within polyhedral theory and combinatorial optimization are abundant, and only a few are discussed in this chapter. When considering lifting techniques, the goal is to achieve facet defining valid inequalities; unfortunately, a common challenge of researchers is to find these inequalities in a reasonable amount of time.

Clearly, being able to find the various sequential simultaneously lifted inequalities would be a valuable tool for researchers. Thus a research topic would be to develop

an algorithm to find many of the $p!$ sequential simultaneously lifted inequalities at one time.

Another option would be to expand the algorithms, or develop new ones, to generate valid inequalities for knapsack problems with multiple constraints. Future research could also extend to developing these concepts for general integer programs with negative constraints.

# Bibliography


[1] Anbil R., Gelman E., Patty B., and Tanga R. (1991), "Recent advances in crew pairing optimization at American airlines," *Interfaces* **21**, 62-74.

[2] Arunapuram, S., K. Mathur and D. Solow (2003). "Vehicle routing and scheduling with full truckloads," *Transportation Science*, **37**, n 2, May 2003, 170-82.

[3] Atamtürk, A., G. Nemhauser and M. Savelsbergh (2000). "Conflict graphs in solving integer programming problems," *European Journal of Operational Research*, **121**, 40-55.

[4] Atamtürk, A. (2003). "On the facets of the mixed-integer knapsack polyhedron," *Mathematical Programming*, **98** (1-3), 145-175.

[5] Atamtürk, A. (2004). "Sequence independent lifting for mixed-integer programming," *Operations Research*, **52** (3), 487-491.

[6] Balas, E., (1975). "Facets of the knapsack polytope", *Mathematical Programming*, **8**, 146-164.

[7] Balas, E and E. Zemel (1978). "Facets of the knapsack polytope from minimal covers," *SIAM Journal of Applied Mathematics*, **34**, 119-148.

[8] Balas, E. and E. Zemel, (1984). "Lifting and complementing yields all the facets of positive zero-one programming polytopes," in *Mathematical Programming, Proceedings of the International Conference on Mathematical Programming*, R.W. Cottle et al., eds., 13-24.

[9] Balas, E. and S. M. Ng (1989). "On the set covering polytope. II, Lifting the facets with coefficients in 0,1,2," *Mathematical Programming*, **45** (1), 1-20.

[10] Balas, E. and M. Fishetti (1993). "Lifting procedure for the asymmetric traveling salesman polytope and a large new class of facets," *Mathematical Programming*, **58** (3), 325-352.

[11] Beasley, J. "The OR Library", http://people.brunel.ac.uk/ mastjjb/jeb/info.html.

[12] Bertsimas, D., C. Darnell and R. Soucy (1999). "Portfolio construction through mixed-integer programming at Grantham, Mayo, Van Otterloo and Company," *Interfaces*, **29**, 1, Jan.-Feb. 1999, 49-66.

[13] Bixby, R., S. Ceria, C. McZeal and M. Savelsbergh. "The MIP-LIB Library," http://www.caam.rice.edu/ bixby/miplib/miplib.html.

[14] Brown, D. and I. Harrower (2004). "A new integer programming formulation for the pure parsimony problem in haplotype analysis," *Algorithms in Bioinformatics. 4th*

*International Workshop, WABI 2004. Proceedings Lecture Notes in Bioinformatics* **3240**, 254-65.

[15] Cho C., D., M. Padberg, and M. Rao (1983). "On the uncapacitated plant location problem. II. Facets and lifting theorems," *Mathematics of Operations Research*, **8** (4), 590-612.

[16] Computational Infrastructure for Operations Research Website (COIN-OR), http://www.coin-or.org/.

[17] Chvátal, V. (1980). "Hard knapsack problems," *Operations Research*, **28(6)**, 1402-1412.

[18] Dahan, X., M. Maza, E. Schost. W. Wu, and Y. Xie (2005). "Lifting techniques for triangular decompositions," *Proceedings of the 2005 International Symposium on Symbolic and Algebraic Computation ISSA '05.*

[19] Dantzig, G., D. Fulkerson, and S. Johnson (1954). "Solution of a large-scale traveling salesman problem," *Operations Research*, **5**, 266-277.

[20] De Farias Jr, I., E. Johnson, and G. Nemhouser (2002). "Facets of the complimentarity knapsack polytope," *Mathematics of Operations Research*, **27** (1), 210-227.

[21] De Simone, C., (1990). "Lifting facets of the cut polytope," *Operations Research Letters* **9** (5), 341-344.

[22] Douglas, K. (2004). *WESP retention report to the College of Engineering*, Kansas State University, Manhattan, KS.

[23] Easton, K., G. Nemhauser and M. Trick (2003). "Solving the traveling tournament problem: A combined integer programming and constraint programming approach," *Practice and Theory of Automated Timetabling IV. 4th International Conference, PATAT 2002*, Selected Revised Papers (*Lecture Notes in Comput. Sci.* Vol.2740), 2003, p 100-9.

[24] Easton, T. and P. Surve (2006), Trees and the linear arrangement problem, Proceedings of Industrial Engineering Research Conference, May 21-23, Orlando, FL.

[25] Easton, T. and K. Chinn (2006), Weighted matchings and the vehicle routing problem Proceedings of Industrial Engineering Research Conference, May 21-23, Orlando, FL.

[26] Easton, T. and K. Hooker, "Simultaneously Lifting Sets of Binary Variables into Cover Inequalities for Knapsack Polytopes," *Discrete Optimization, Special Issue: In Memory of George B. Dantzig*, **5**(2) May 2008, 254-261.

[27] Easton, T., K. Hooker and E. K. Lee, (2003) "Facets of the Independent Set Polytope," *Mathematical Programming, Series B* **98**(1-3), 177-199.

[28] Felici, G., and C. Gentile (2003). "Zero-lifting for integer blocks structured problems," *Journal of Combinatorial Optimization*,**7** (2), 161-167.

[29] Ferreira, C., C. de Souza and Y. Wakabayashi (2002). "Rearrangement of DNA fragments: A branch-and-cut algorithm," *Discrete Applied Mathematics*, **116**, (1-2), 15 Jan. 2002, 161-77.

[30] Gomory, R. E., (1958). "Outline of an algorithm for integer solutions to linear programs," *Bulletin of the American Mathematical Society*, **64**, 275-278.

[31] Gomory, R. (1960). "Solving linear programming problems in integers," *Combinatorial Analysis*, R.E. Bellman and M. Hall, Jr. eds., American Mathematical Society 211-216.

[32] Gomory, R. (1963). "An algorithm for integer solutions to linear programs," *Recent Advances in Mathematical Programming*, R. Graves and P. Wolfe, eds. McGraw-Hill 269-302.

[33] Gomory, R. (1963). "An all-integer programming algorithm," *Industrial Scheduling*, J.F. Muth and G. Thompson, eds. Prentice-Hall 193-206.

[34] Gomory, R. (1969). "Some polyhedra related to combinatorial problems," *Linear Algebra and its Applications*, **2**, 451-558.

[35] Gomory, R., (2007). "The atoms of integer programming ' *Annals of Operations Research*, **149**, 99-102.

[36] Gomory, R. and E. Johnson (1972). "Some continuous functions related to corner polyhedra," *Mathematical Programming*, **3**, 23-85.

[37] Gomory, R. and E. Johnson (1973). "The group Problem and subadditive functions," *Mathematical Programming*, T. Hu and S. Robinson, eds. Academic Press, 157-184.

[38] Gu, Z., G. Nemhauser, and M. Savelsbergh (1998). "Lifted cover inequalities for 0-1 integer programs: computation," *INFORMS Journal on Computing*, **10** (4), 427-437.

[39] Gu, Z., G. Nemhauser, and M. Savelsbergh (1999). "Lifted cover inequalities for 0-1 integer programs: complexity," *INFORMS Journal on Computing*, **11** (1), 117-123. 109-121.

[40] Gu, Z., G. Nemhauser, and M. Savelsbergh (2000). "Sequence independent lifting in mixed integer programming," *Journal of Combinatorial Optimization*, **4**, 109-129.

[41] Gutierrez, Talia, (2007) "Lifting general integer variables," *MS Thesis*, Department of Industrial and Manufacturing Systems Engineering, Kansas State University.

[42] Hammer, P., E. Johnson, and U. Peled, (1975). "Facets of regular 0-1 polytopes," *Mathematical Programming*, **8**, 179-206.

[43] Hooker, K. and T. Easton, (2007) "Using Hyperstars to Create Facial Defining Inequalities of General Binary Integer Programs," *The International Journal of Operations Research*, **4** (3), 138-145.

[44] Hooker, Kevin, (2005) "Hypergraphs and integer programming polytopes, *Ph.D. Dissertation*, Department of Industrial and Manufacturing Systems Engineering, Kansas State University.

[45] Hunsaker, B. and C. Tovey (2004). "Simple lifted cover inequalities and hard knapsack problems," *Technical Report: Industrial Engineering, University of Pittsburgh*, Pittsburgh, PA 1-13.

[46] Huschka, Bryce, (2007) "Finding adjacent facet-defining inequalities," *MS Thesis*, Department of Industrial and Manufacturing Systems Engineering, Kansas State University.

[47] Karp, R. (1972). "Reducibility among combinatorial problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, eds., Plenum Press, New York 85-103.

[48] Kaufman, D., J. Nonis, and R. Smith (1998). "A mixed integer linear programming model for dynamic route guidance," *Transportation Research, Part B (Methodological)*, **32B** (6), Aug. 1998, p 431-40.

[49] Kayarat, Dheeraj (2005) "The quadratic polyhedral clustering algorithm A new method To cluster microarray data," *MS Thesis*, Department of Industrial and Manufacturing Systems Engineering, Kansas State University.

[50] Koster, A., S. Hoesel, and A. Kolen (1998). "The partial constraint satisfaction problem: facets and lifting," *Operations Research Letters*, **23** (3), 89-98.

[51] Land, A.H. and A.G. Doig (1960). "An automatic method for solving discrete programming problems," *Econometrica*, **28**, 497-520.

[52] Lee, E. and M. Zaider, (2000). "Mixed integer programming approaches to treatment planning for brachytherapy – Application to permanent prostate implants," *Annals of Operations Research, Optimization in Medicine*, 2000, 147-163.

[53] Lee, E., T. Fox and I. Crocker, (2003) "Integer programming applied to intensity-modulated radiation treatment planning optimization," *Annals of Operations Research, Optimization in Medicine*, 2003, 119: 165-181.

[54] McAdoo, Michael, (2007) "Three set inequalities in integer programming," *MS Thesis*, Department of Industrial and Manufacturing Systems Engineering, Kansas State University.

[55] Nemhauser, G. and L. Wolsey, (1988). *Integer and combinatorial optimization*, John Wiley and Sons, New York.

[56] Nemhauser, G. L. and P. H. Vance, (1994). "Lifted cover facets of the 0-1 knapsack polytope with GUB constraints," *Operations Research Letters*,**16** (5), 255-263.

[57] Padberg, M. (1973). "On the facial structure of set packing polyhedra," *Mathematical Programming*, **5**, 199-215.

[58] Park, K. (1997). "Lifting cover inequalities for the precedence-constrained knapsack problem," *Discrete Applied Mathematics*, **72** (3), 219-241.

[59] Pinto, R. and B. Rustem (1998) "Solving a Mixed-Integer Multiobjective Bond Portfolio Model Involving Logical Conditions," *Annals of Operations Research*, **81**, 1998, 497-513.

[60] Richard J., I. De Farias, and G. Nemhauser (2003). "Lifted inequalities for 0-1 mixed integer programming: Superlinear lifting," *Integer Programming*, **98** (1-3), 115-143.

[61] Ruiz, R., C. Maroto and J. Alcaraz (2004). "A decision support system for a real vehicle routing problem," *European Journal of Operational Research*, **153** (3), 16 March 2004, 593-606.

[62] Santanu, S. D. and P. R. Jean-Philippe, (2006). "Linear programming based lifting and its application to primal cutting plane algorithms," School of Industrial Engineering, Purdue University.

[63] Sharma, Kamana, (2007) "Simultaneously lifting sets of variables in binary knapsack problems," *MS Thesis*, Department of Industrial and Manufacturing Systems Engineering, Kansas State University.

[64] Shebalov, S. and D. Klabjan, (2006) "Sequence independent lifting for mixed integer programs with variable upper bounds," *Mathematical Programming*, **105** (2-3), 523-561.

[65] Subramanian, R., Scheff R., Quillinan J., Wiper D. S., and Marsten R. (1994) "Coldstart: Fleet Assignment at Delta Air Lines." *INTERFACES*, **24**, January-February 1994, 104-120.

[66] The CPLEX Solver on ILOG's Home Page, http://www.ilog.com/.

[67] Toth, P. (1997). "An exact algorithm for the vehicle routing problem with backhauls," *Transportation Science*, **31** (4), Nov. 1997, 372-85.

[68] Urban, T. (2003). "Scheduling sports competitions on multiple venues," *European Journal of Operational Research*, **148** (2), 16 July 2003, 302-11.

[69] Wolsey, L.A. (1975). "Faces for a linear inequality in 0-1 variables," *Mathematical Programming*, **8**, 165-178.

[70] Wolsey, L.A. (1976). "Facets and strong valid inequalities for integer programs," *Operations Research*, **24**, 367-372.

[71] Wolsey, L.A. (1977). "Valid inequalities and superadditivity of 0/1 integer programs," *Mathematics of Operations Research*, **2**, 66-77.

[72] Zemel, E. (1978). "Lifting the facets of 0-1 polytopes" *Mathematical Programming*, **15**, 268-277.

[73] Zemel, E. (1989). "Easily computable facets of the knapsack polytope," *Mathematics of Operations Research*, **14**, 760-764.