GENERATING AN ORIGINAL CUTTING-PLANE ALGORITHM IN THREE SETS (GO CATS)

by

ANDREW WILLIAM HARRIS

B.S., Kansas State University, 2010

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

2010

Approved by:

Major Professor

Dr. Todd Easton

ABSTRACT

Integer programs (IP) are a commonly researched class of problems used by governments and businesses to improve decision making through optimal resource allocation and scheduling. However, integer programs require an exponential amount of effort to solve and in some instances a feasible solution is unknown even with the most powerful computers.

There are several methods commonly used to reduce the solution time for IPs. One such approach is to generate new valid inequalities through lifting. Lifting strengthens a valid inequality by changing the coefficients of the variables in the inequality. Lifting can result in facet defining inequalities, which are the theoretically strongest inequalities.

This thesis introduces the Cutting-plane Algorithm in Three Sets (CATS) that can help reduce the solution time of integer programs. CATS uses synchronized simultaneous lifting to generate a new class of previously undiscovered valid inequalities. These inequalities are based upon three sets of indices from a binary knapsack integer program, which is a commonly studied integer program. CATS requires quartic effort times the number of inequalities generated. Some theoretical results describe easily verifiable conditions under which CATS inequalities are facet defining.

A small computational study shows CATS obtains about an 8.9% percent runtime improvement over a commercial IP software. CATS preprocessing time is fast and requires an average time of approximately .032 seconds to perform. With the exciting new class of inequalities produced relatively quickly compared to the solution time, CATS is advantageous and should be implemented to reduce solution time of many integer programs.

Dedication

My work is dedicated to

my father, Roger

my mother, Barbara

my brother, Paul

and all those whom I've been blessed to call my friends and brothers during my time at Kansas State University.

And finally, my work is ultimately dedicated to the Lord our God, through Whom all of this is possible.

Acknowledgments

Many gave their time and effort to make this thesis a success. I would like to sincerely thank

Dr Todd Easton, the inspiration, the perspiration, a dynamic and patient mentor

Dr Joseph Harner and Dr John Wu, for serving on my committee

The IMSE Faculty, for their wonderful professional instruction

The IMSE Staff, for brightening my day

And finally, my classmates in Industrial Engineering during the past three years,

who together with me endured hardships and made learning a joyful experience.

Contents

List of Figures

List of Tables

1 Introduction

This thesis introduces the Cutting-plane Algorithm in Three Sets (CATS) that can help reduce the solution time of integer programs. Integer programs (IP) are a commonly researched class of problems from the field of discrete optimization. On a daily basis, governments and businesses improve decision making by solving various types of integer programs.

With the recognition of IP's importance, the proliferation of integer programming has spread to many business sectors over the last 60 years. A commonly thought of example involves the transportation industry [2, 25, 34, 39] with its truck routing problems. IP has been used to solve financial problems, such as capital budgeting decisions [18, 23] and financial portfolios [8, 33] to maximize earnings. In the medical profession, genetic research [10, 17] and radiation treatments [28, 29] contain applications of integer programming. There are applications of IP in the timber [11] and airline industries [1, 37] that save millions of dollars annually. Even sports leagues, with their revenue dependence on marketing contracts and fan support, use integer programming to develop the best schedules for all teams [14, 40, 41].

Surprisingly, the average person unknowingly tries to solve an IP each time he goes to the grocery store. Suppose a person with $100 goes to the store. Each item in the store has both a cost in dollars and an anticipated benefit. What should be purchased to maximize the person's benefit without exceeding the $100? This small grocery example illustrates an important IP application critical to this thesis: the Knapsack Problem (KP).

KP is an application of integer programming where all the variables within the problem

are binary. This famous example draws an analogy to a hiker preparing for an excursion. He must decide to either bring along an item in his knapsack or omit the item. The items each have a specific benefit and a specific weight. The hiker cannot bring a group of items that violate a total weight.

In a complex industrial problem where an item may cost millions of dollars, the decision could make or break the company. KPs are especially helpful when modeling scheduling [3, 5, 26, 32] and capital budgeting problems [6, 22, 31], and this model is also useful for portfolio allocations [13, 30, 35, 45].

Formally, an integer program is defined as:

$$\begin{aligned} Z^{IP} = \text{Maximize } & c^T x \\ \text{subject to } & Ax \leq b \\ & x \in \mathbf{Z}_+^n. \end{aligned}$$

Integer programs are $\mathcal{NP}$-Hard [24]. Unless $\mathcal{P} = \mathcal{NP}$, solving IP problems require exponential time. In a problem with binary variables, exhaustively enumerating all possible solutions to the IP results in 2^n evaluations. In many IP instances, an IP can run on a computer for years before finding a solution, let alone the optimal solution.

The most common way to solve integer programs is using the technique called the Branch and Bound Algorithm (BBA), which utilizes the optimal solution, z^{*LR}, and x^{*LR}, from the linear relaxation. The linear relaxation of an integer program merely removes the integer requirement. Thus, the optimal solution may have fractional variables. In such a situation, BBA creates two nodes or children. One of the children adds the constraint that a single

fractional variable is less than or equal to the floor of the value of that variable in the optimal linear relaxation solution. The other adds the constraint that is greater than or equal to the ceiling of this value. Doing this enough times enables the exhaustive enumeration of all integer points in a bounded region and gives the optimal integer solution, z^{*IP}, and feasible solution, x^{*IP}.

A common method to try to reduce BBAs solution time is through the use of cutting planes, also known as valid inequalities. Cutting planes are linear inequalities that are valid for all the feasible points. In integer programming, cutting planes intersect some of the linear relaxation space. The best cutting planes for the space are known as facet defining inequalities and are the theoretically strongest of all valid inequalities.

Lifting seeks to increase the effectiveness of the inequality by changing the coefficient of the variables. A valid inequality that is not facet defining can frequently be made facet defining through lifting.

1.1 Motivation

In 2009, Bolton developed a Synchronized Simultaneous Lifting (SSL) algorithm that lifts two sets of variables [9]. Her computational results showed a significant improvement in processing time of IPs on a commercial IP solver. Additionally in 2009, Kubik developed a pseudopolynomial time algorithm for knapsack problems to sequentially lift multiple sets simultaneously into the knapsack constraint [27]. A natural question is: If synchronized lifting three sets simultaneously is effective, is it possible for SSL to be extended to three sets? Early research using manual calculations and hand-made graphs showed promise of

new effective cutting planes.

1.2 Contributions

The contribution of this thesis is the Cutting-plane Algorithm in Three Sets (CATS). CATS combines parts of Bolton's and Kubik's algorithms in a new advanced geometric surfacing procedure that finds valid cutting planes. By increasing the number of sets from two to three, many more valid inequalities are created from three sets as compared with two sets, thereby increasing the algorithm's potential effectiveness.

The input to CATS is a knapsack constraint and three sets of mutually exclusive indices. Cutting planes are then formed using the outer most points of a related three dimensional space. These cutting planes build off one another, and each new plane is formed using information from the previous plane. The analogy of the framework being built over a large sports arena illustrates how the algorithm seeks out the adjacent facets around the edge of the integer space.

The contributions of this thesis lie in CATS' ability to systematically define and find adjacent facets on the edge of the feasible integer space. CATS generates these valid inequalities in a runtime of $O(n^4|T|)$, where $|T|$ is the number of inequalities generated. CATS is able to produce this efficient running time by utilizing adjacent facet points to compute valid inequalities

The inequalities generated by CATS are shown to be a new class of previously undiscovered inequalities. Furthermore, a theoretical result provides easily checkable conditions when CATS cuts are facet defining. Thus, these inequalities are a new class of facet defining

inequalities for the knapsack polyhedron and thus should be useful in practice.

A small computational study of CATS shows an improved solution time for certain IPs. Two sets of 20 randomly generated problems showed an average improvement of 8.9% compared to traditional CPLEX 10.0, a commercial integer programming software. This amounted to an improvement of nearly 50 minutes on the 7.5 hour total runtime. Some smaller problems solved over 50% faster then CPLEX. Thus, CATS gives promise of reducing the runtime in large integer programs.

1.3 Outline

As described previously, integer programming is an important area of research for efficient use of resources and sound decision making. The goal of this thesis is to extend some of IP's theoretical development, as well as orient the reader to the field of integer programming. The remainder of this thesis focuses on the fundamentals of IP and CATS itself.

Chapter 2 provides the IP definitions and terminology applicable to this thesis. The Knapsack Problem is described in depth, and the benefits of cutting planes are discussed. Various types of lifting methods and combinations of lifting methods are explained as well. Examples of several of these lifting methods are shown.

Chapter 3 describes the Cutting-plane Algorithm in Three Sets. This chapter defines the terminology of CATS with an explanation of the algorithm's steps. Next, the proofs for inequality validity and termination as well as the pseudocode further illustrate CATS to the reader. A small example with diagrams help to convey the major concepts of CATS.

A summary of the computational procedure and results are presented in chapter 4. The types of problems generated and analyzed are described, as well as data to support that these inequalities are effective in reducing the runtime of certain randomly generated integer programs.

Finally, chapter 5 provides a conclusion of CATS and its computations results. There are several possible future research topics resulting from this work. These are described as well.

2 Background Information

This chapter introduces the mathematical and integer programming background to understand this thesis. This requires the understanding of not only linear algebra concepts, but also the knowledge of various types of lifting and the importance of producing facet defining inequalities. Through this chapter's descriptions, a person can become acquainted with the concepts well enough to understand both the logic and benefits of CATS. An excellent technical reference for additional IP information is Nemhauser and Wolsey [30].

The first section outlines some basic concepts of integer programming. Polyhedrons and cutting planes are then formally introduced. Cover inequalities and their application to the Knapsack Problem are explained. Finally, there are several sections describing types of lifting.

2.1 Integer Programming

To begin, it is paramount to understand the basic definitions of integer programming. Formally, an IP is defined by

$$z^{IP} = \text{Maximize } \ c^T x$$

$$\text{subject to } Ax \leq b$$

$$x \in \mathbf{Z}^n_+$$

where $c \in \mathbf{R}^n$ represents the objective coefficients, $A \in \mathbf{R}^{m \times n}$ and $b \in \mathbf{R}^{m \times 1}$ describes the constraints. The feasible points of an integer program are denoted by P and so $P = \{x \in \mathbf{Z}^n_+ : Ax \leq b\}$. Here the set of indices of the x variables is denoted as $N = \{1, ..., n\}$.

Since one cannot typically solve an integer program quickly, the integer restrictions of the IP are relaxed. This creates a linear program, which is called the linear relaxation. The formal definition of the linear relaxation is:

$$z^{LR} = \text{ Maximize } c^T x$$

$$\text{subject to } Ax \leq b$$

$$x \in \mathbf{R}^n_+.$$

The feasible solution area for the linear relaxation is denoted P^{LR}, defined as $\{x \in \mathbf{R}^n_+ : Ax \leq b\}$. The optimal solution, z^{*LR} from x^{*LR}, of the linear relaxation is found using traditional linear programming algorithms [12]. Clearly $P \leq P^{LR}$, thus $z^{IP} \leq z^{LP}$.

2.1.1 Polyhedrons, Cutting Planes and Facets

Convexity is a fundamental topic in mathematical optimization. A set S is convex if, and only if, $x \in S$ where $x = \sum_{i=1}^n \lambda_i x^i$ for some finite set of points $\{x^i : i \in S : i = 1, ..., n\}$ and some $\lambda \in \mathbf{R}^n_+$ with $\sum_{i=1}^n \lambda_i = 1$. Given a set $S \subseteq \mathbf{R}^n$ the intersection of all of the convex sets that contain S is called the convex hull of S and is denoted by S^{ch}. This means that all the points on a line connecting any two points in S is in S^{ch}.

Any single linear inequality restricts the feasible region to either above or below the hyperplane. This is known as a half-space. Formally, a half space is defined as $\{x \in \mathbf{R}^n : \sum_{i=1}^n \alpha_i x_i \leq \beta\}$. The feasible region can be further defined by adding more inequalities. The intersection of finitely many half spaces is called a polyhedron, and a bounded polyhedron is a polytope.

There are two important polyhedra associated with integer programming research. First, P^{LR} is defined by the linear relaxation bounds of the problem. The extreme points for P^{LR} can be integer or non-integer. Second, P^{ch} is defined by the original linear bounds and the integer constraints, and P^{ch} has integer extreme points. One common IP research area is to add inequalities to P^{LR} in an attempt to convert it to P^{ch}.

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 $x' \in P^{ch}$ or equivalently for every $x' \in P$. A cutting plane is a valid inequality that removes some of P^{LR}. Clearly there are infinitely many valid inequalities and some are useful, while others can adversely affect the solution time of an integer program.

Theoretically, the usefulness of a valid inequality can be measured in terms of its induced dimension on P^{ch}. The dimension of the space is defined by the total number of linearly independent vectors. However, P has no feasible vectors and so the dimension of P^{ch} is calculated as the maximum number of affinely independent points minus one. The points $x^1, ..., x^q$ in R^n are affinely independent if, and only if, the unique solution to $\sum_{i=1}^{q}\lambda_i x^i = 0$ and $\sum_{i=0}^{q}\lambda_i = 0$ is $\lambda_i = 0$ for all $i = 1, ..., q$.

The induced points of an inequality on P^{ch} is called a face. Every valid inequality $\Sigma_{j=1}^{n}\alpha_j x_j \leq \beta$ defines a face $F \subseteq P^{ch}$ that takes the form $F = \{x \in P^{ch} : \Sigma_{j=1}^{n}\alpha_j x_j = \beta\}$. If $F \neq \emptyset$, then F supports P^{ch}. The strength of a face depends upon its dimension relative to P^{ch}. The higher dimension of the face, as long as it is not equal to that of P^{ch}, the more likely it is to be useful.

Facet defining inequalities are the strongest of all inequalities. A facet defining inequality has a face of dimension one less than the dimension of P^{ch}. Including all facet defining

inequalities completely defines P^{ch}. Thus, all the extreme points are integer. In such a situation, an integer program may be solved as a linear program.

The following example depicts these concepts.

Example

Consider the following integer program:

Maximize $x_1 + 2x_2$

Subject to $5x_1 + 4x_2 \leq 20$

$3x_1 + 5x_2 \leq 18$

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

Figure 2.1.1 provides a graphical view of this IP. The first constraint, $5x_1 + 4x_2 \leq 20$, passes through points $(0, 5)$, C and D. The second constraint, $3x_1 + 5x_2 \leq 18$, passes through the points A, B, C and $(6, 0)$. The linear relaxation of the IP is defined by these two constraints, the x_1 and x_2 axes. The large circles represent P, the feasible integer points.

This illustration shows just how a new cutting plane may be beneficial in removing part of the linear relaxation space. As is shown, a new inequality, $x_1 + x_2 \leq 4$, is inserted into the linear relaxation. It does so without removing any feasible integer points. Therefore, $x_1 + x_2 \leq 4$ is a valid inequality. The new cutting plane removes part of the linear relaxation located within the BCD triangle. This graph provides a better definition of the extreme integer points of the convex hull by passing through points $(0, 4)$, B and D.

As stated earlier, a facet defining inequality is an inequality with dimension of one less than the convex hull it is defining. The dimension of P^{ch} can be bounded from below using

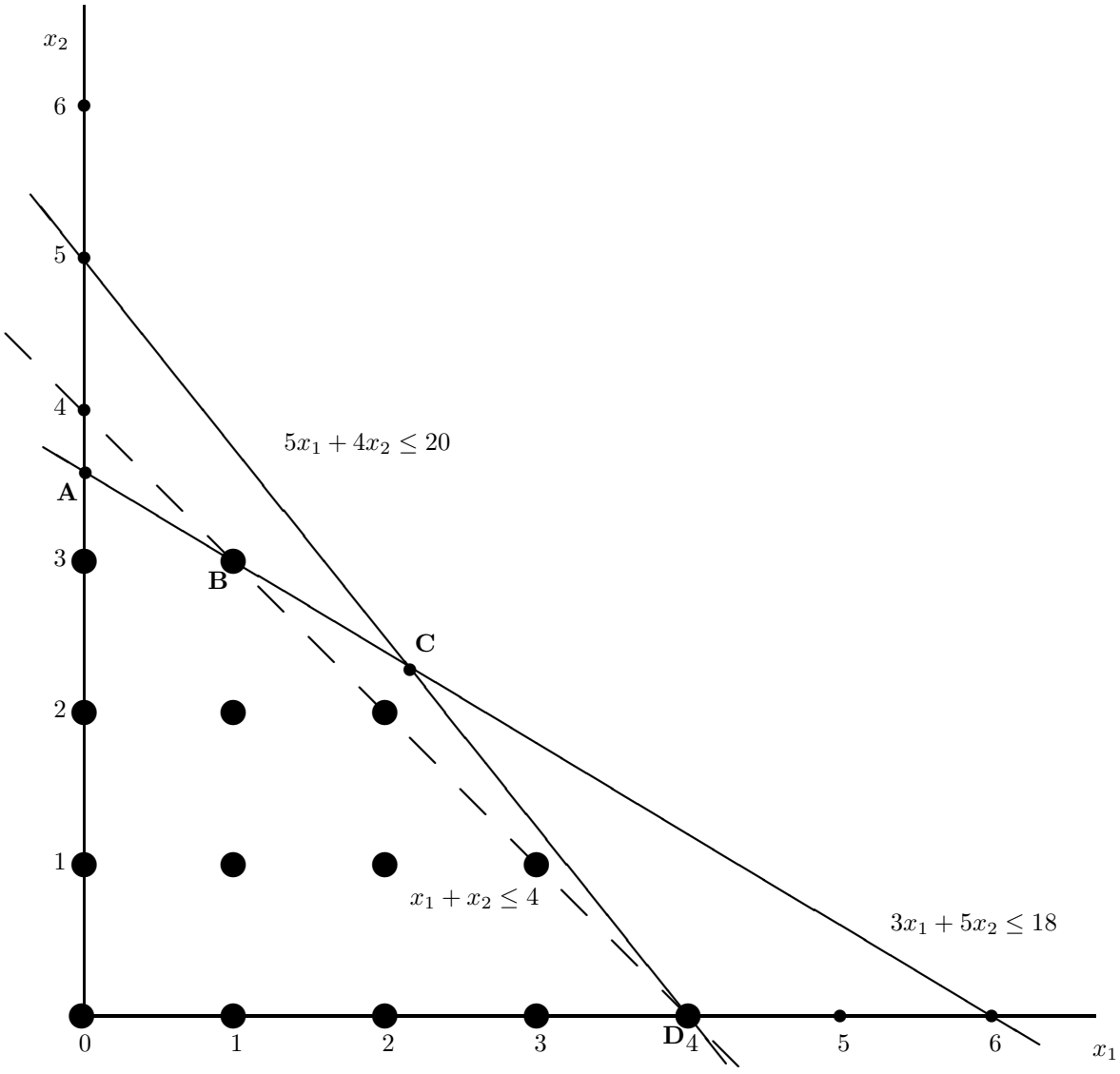

Figure 1: Cutting plane method in example 2.1.1

the maximum number of affinely independent points in the space minus one. The points $(0,0)$, $(0,1)$ and $(1,0)$ are affinely independent and feasible. The number of variables in the example, two, bounds the dimension from above, and so the dimension of P^{ch} is two. The dimension of the facet is found to be one using the affinely independent points of $(1,3)$ and $(4,0)$. Additionally, it can clearly be seen that the new inequality does not violate any integer point. Thus, $x_1 + x_2 \leq 4$ is a valid inequality. The other facet defining inequalities of P^{ch} are $x_2 \leq 3$, $x_1 \geq 0$ and $x_2 \geq 0$.

2.2 The Knapsack Problem

A well-known class of integer programming problems important to this thesis is the Knapsack Problem (KP). When preparing for the mountain trail, the hiker may choose from a total number of items n and must decide to include an item j or omit the item from his pack. A nonnegative weight a_j and a benefit c_j are assigned to each item, and he is limited by the total weight he can carry, b. The hiker tries to maximize his benefit.

Formally, an IP formulation for The Knapsack Problem begins by setting $x_j = 1$ if item j is taken and $x_j = 0$ else.

Maximize $\sum_{i=1}^{n} c_i x_i$

subject to $\sum_{i=1}^{n} a_i x_i \leq b$

$x_i \in \mathbf{B}$ for all $i = 1, 2, ..., n.$

Let P_{KP} be the set of feasible points of a KP problem, $P_{KP} = \{x \in \mathbb{B}^n : \sum_{i=1}^{n} a_i x_i \leq b\}$. Now let P_{KP}^{ch} be the intersection of every convex set that contains, $P_{KP}^{ch} = conv(P_{KP})$.

Without loss of generality, assume that the a_i's are sorted in descending order; if $i, j \in N$ and $i < j$, then $a_i \geq a_j$. Furthermore, $a_1 \leq b$ or $x_1 = 0$ in all feasible solutions and can therefore be removed from the KP instance. Consequently, P_{KP}^{ch} is full dimensional using 0 and e_i where e_i is the i^{th} identity point for $i = 1, ..., n$.

The applicability of KP research to general IP problems makes this area of research highly attractive. Any single binary integer programming constraint can be converted into a knapsack instance through a simple transformation. If the constraint is an $=$ constraint, two constraints are formed, $\leq$ and $\geq$. Any $\geq$ constraint is multiplied through by a -1. If there exists an $a_i < 0$, then x_i is replaced with $1 - x_i'$. Thus, knowledge of cutting planes for knapsack instances can be applied to any single binary integer programming constraint.

The following example illustrates KP.

Example 2.1 A hiker considers taking 16 items on a hiking trip. The hiker has assigned each item with a benefit and a weight as shown in Table 1. The weights are listed in 1/2 pound units and the hiker can carry 47 lbs (or 94 units).

Object	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Benefit	25	29	20	13	17	12	11	15	13	14	19	16	7	6	9	8
Weight	31	30	28	25	20	20	19	18	18	17	17	15	14	14	13	12

Table 1: Benefit and weight of items that may be taken in the knapsack

An associated model is:

Maximize $\quad 25x_1 + 29x_2 + 20x_3 + 13x_4 + 17x_5 + 12x_6 + 11x_7 + 15x_8 + 13x_9$

$$+14x_{10} + 19x_{11} + 16x_{12} + 7x_{13} + 6x_{14} + 9x_{15} + 8x_{16}$$

$$\text{subject to} \quad 31x_1 + 30x_2 + 28x_3 + 25x_4 + 20x_5 + 20x_6 + 19x_7 + 18x_8 + 18x_9$$

$$+17x_{10} + 17x_{11} + 15x_{12} + 14x_{13} + 14x_{14} + 13x_{15} + 12x_{16} \leq 94.$$

$$x_j \in \{0, 1\}, j \in \{1, ..., 16\}.$$

Solving this with an integer program shows that the hiker achieves a maximum benefit of 89 by taking items $1, 2, 11$ and 12 (25+29+19+16=89). The hiker should carry a total weight of 46.5 lbs.

2.2.1 Covers

Finding facet defining inequalities in two dimensions is easy. In higher dimensions, it is much more difficult. A cover is a set of indices from a binary knapsack constraint such that setting all x_j equal to one is infeasible (heavier than the load the hiker can carry). Covers are primarily used to find an initial valid inequality and later the inequality is strengthened through lifting. The use of cover inequalities in KP is important when seeking to create facet defining inequalities.

Formally, $C \subseteq N$ is a cover if and only if $\Sigma_{j \in C} a_j > b$. A minimal cover is a cover such that when one indice is removed from the set, the set is no longer a cover. In other words, $\Sigma_{j \in C \setminus \{k\}} a_j \leq b$ for each $k \in C$.

Each cover (whether minimal or not) defines a cover inequality. Cover inequalities are valid and take the form $\Sigma_{j \in C} x_j \leq |C| - 1$. This is a valid inequality because the sum of all of the coefficients $\Sigma_{j \in C} a_j$ is greater than the maximum amount allowed by the constraint;

thus, at least one variable must be set to zero in every feasible solution.

Given a cover, an extended cover is $E(C) = C \cup \{i \in N : a_i \geq a_j \text{ for all } j \in C\}$ and the extended cover inequality takes the form $\Sigma_{j \in E(C)} x_j \leq |C| - 1$. The following example depicts these concepts.

Example 2.2 In looking back at the example Knapsack Problem from Example 2.1, the single constraint is: $31x_1 + 30x_2 + 28x_3 + 25x_4 + 20x_5 + 20x_6 + 19x_7 + 18x_8 + 18x_9$

$$+17x_{10} + 17x_{11} + 15x_{12} + 14x_{13} + 14x_{14} + 13x_{15} + 12x_{16} \leq 94$$

For this problem, a cover is $C = \{9, 10, 11, 12, 13, 14, 15, 16\}$ because $18 + 17 + 17 + 15 + 14 + 14 + 13 + 12 = 120 \geq 94$. This is not a minimal cover, as indice 9 may be removed and the set remains infeasible. A minimal cover is $C = \{10, 11, 12, 13, 14, 15, 16\}$ because $17+17+15+14+14+13+12 = 102 \geq 94$. Notice that when any of the indices are removed from C, the set becomes feasible and is no longer a cover. Therefore, the cover inequality generated by the minimal cover $C = \{10, 11, 12, 13, 14, 15, 16\}$ is $x_{10} + x_{11} + x_{12} + x_{13} + x_{14} + x_{15} + x_{16} \leq 6$. Its extended cover inequality is $x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} + x_{14} + x_{15} + x_{16} \leq 6$. This has the dimension of at least six due to the points in Figure 2 below.

As shown, seven affinely independent points for the minimal cover inequality yield a dimension of six. The cyclical permutation of any six of the seven points is shown. Cyclical permutations are useful to determine the strength of inequalities and can help show an inequality is facet defining. For this inequality to be facet defining, 16 affinely independent points are needed. The next section details how facet defining inequalities may be found

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	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	0	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	0	1	1	1
1	1	1	1	0	1	1
1	1	1	1	1	0	1
1	1	1	1	1	1	0

Figure 2: Affinely independent points for knapsack cover

through lifting.

2.3 Lifting

The purpose of lifting inequalities in integer programming is to create stronger inequalities from a valid inequality in a restricted space. The basic idea of a restricted space is that a subset of the variables, say E, are forced to a fixed value, k_i for each $i \in E$. Formally, let $P^{ch}_{E,K} = conv\{x \in P : x_i = k_i \text{ for all } i \in E\}$ where $k_i \in \mathbf{Z}$ and $K = (k_1, k_2, ..., k_{|E|})$.

In formal terms, suppose $\Sigma_{i\in E}\alpha_i x_i + \Sigma_{i\in N\setminus E}\alpha_i x_i \leq \beta$ is a valid inequality in $P^{ch}_{E,K}$ where $E \subset N$. Lifting attempts to create a valid inequality of P^{ch} that takes the form $\Sigma_{i\in E}\alpha'_i x_i + \Sigma_{i\in N\setminus E}\alpha_i x_i \leq \beta'$. Lifting was first developed by Gomory [19] and is used to strengthen inequalities by increasing the dimension of the cutting plane. The various types of lifting are up lifting or down lifting, exact or approximate lifting, sequential or simultaneous lifting, and

single or synchronized lifting. An example of a lifting method would be single, approximate, simultaneous up lifting, which is one of the most studied classes of lifting.

2.3.1 Up vs. Down Lifting

Up lifting is the most common lifting technique and assumes all variables being lifted have no initial coefficient $\alpha_i = 0$ for all $i \in E$, or $K = (0, 0, ..., 0)$. This thesis uses up lifting to create new inequalities. For simplicity, let P_E^{ch} be defined as $P_{E,(0,0,...,0)}^{ch}$. Down lifting assumes that all of the k_i's are set to the upper bound of x_i for all $i \in E$. To date, down lifting has only been done sequentially.

2.3.2 Exact vs. Approximate Lifting

Exact lifting seeks to increase the α' and/or decrease the B' coefficients as much as possible and still maintain a valid inequality. Any increase in the lifted coefficient on the left hand side or a decrease in the right hand side would result in an invalid inequality. This type of lifting frequently requires the exact solution to an integer program, which increases its runtime and makes some instances so computationally challenging that the lifting problem takes longer to solve than the original problem. There is some research into reducing the computation time to polynomial time on certain IP instances, such as KP [7, 15, 35].

Approximate lifting provides an alternative to the computational intensity of exact lifting. Researchers have developed approximate lifting methods that quickly generate coefficients that maintain a valid inequality, but it still can be improved. Thus, the practitioner must frequently decide about the tradeoff between precise solutions and a fast processing time.

Some approximate lifting research includes sequential lifting [5] and sequence independent lifting [4, 20, 36, 43].

2.3.3 Sequential vs. Simultaneous Lifting

Two major types of lifting are sequential and simultaneous lifting. The main difference between these types of lifting is the size of E. In sequential lifting, $|E| = 1$, meaning the variables are lifted one at a time. Simultaneous lifting lifts more than one variable at a time, so $|E| \geq 2$.

2.3.4 Sequential Lifting

Sequential lifting seeks to modify the coefficients of the variables individually. The order in which the variables are lifted into the problem may vary, and this results in different inequalities being produced from different orders. The sequential up lifting algorithm 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 $\alpha_1 x_1 + \Sigma_{j=2}^{n} \alpha_j x_j \leq \beta$ for P^{ch}. Several individuals have performed research on sequential up lifting [5, 7, 42]

To obtain an exact sequential uplifted coefficient for a binary x_i, the following procedure is followed. First, the integer program for sequential up lifting of binary variables is solved.

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

Once the optimal solution is found, $\alpha_1 = \beta - z^*$. Wolsey showed that the dimension of the inequality in P^{ch} increases by at least one over P_1^{ch} for each sequentially lifted variable.

Example 2.3

Reconsider the knapsack polytope from Example 2.1. Observe that $C = \{10,11,12,13,14,15,16\}$ is a cover. The cover inequality is $x_{10} + x_{11} + x_{12} + x_{13} + x_{14} + x_{15} + x_{16} \leq 6$. To sequentially up lift x_1, solve the following IP:

$$z^* = \text{Maximize} \quad x_{10} + x_{11} + x_{12} + x_{13} + x_{14} + x_{15} + x_{16}$$

$$\text{subject to} \quad 31x_1 + 30x_2 + 28x_3 + 25x_4 + 20x_5 + 20x_6 + 19x_7 + 18x_8 + 18x_9$$

$$+17x_{10} + 17x_{11} + 15x_{12} + 14x_{13} + 14x_{14} + 13x_{15} + 12x_{16} \leq 94$$

$$x_1 = 1$$

$$x_i \in \{0, 1\},\ i = 1, ..., 16$$

The solution z^* to the above integer program is $z^* = 4$. This means that $\alpha_1 = \beta - z^*$, or $\alpha_1 = 6-4 = 2$. The resulting valid inequality is $2x_1+x_{10}+x_{11}+x_{12}+x_{13}+x_{14}+x_{15}+x_{16} \leq 6$.

To lift x_9, solve $z^* = \text{Maximize}\ \ 2x_1+x_{10}+x_{11}+x_{12}+x_{13}+x_{14}+x_{15}+x_{16}+x_{17}+x_{18}+x_{19}$ subject to $31x_1 + 30x_2 + ... + 12x_{16} \leq 94$ and $x_9 = 1$. The solution value is $z^* = 5$, which means that $\alpha_9 = \beta - z^* = 6 - 5 = 1$. The resulting valid inequality is $2x_1 + x_9 + x_{10} + x_{11} + x_{12} + x_{13} + x_{14} + x_{15} + x_{16} \leq 6$.

When the process is repeated, for $x_8, x_7, x_6, x_5, x_4, x_3$, and x_2, the final sequentially lifted inequality is $2x_1+2x_2+2x_3+x_4+x_5+x_6+x_7+x_8+x_9+x_{10}+x_{11}+x_{12}+x_{13}+x_{14}+x_{15}+x_{16} \leq 6$. This inequality is facet defining. The points in Figure 3 show that this inequality is facet defining:

1	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	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	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	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	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1
0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1
0	0	0	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	0	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	0	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

Figure 3: Affinely independent points for sequentially lifted inequality

Notice the cyclical permutation that allows for the inequality to reach the required number of affinely independent points, due to the minimal cover. Observe how the x^* solution of the lifted inequality IP is the point used to increase the dimension of the face.

The lifted inequality is determined by the order the variables are sequentially lifted. The variables here could have been lifted in the order $x_1, x_2, ..., x_9$, or any unique combination of this group of variables. In this example, there is a total of nine factorial, $9! = 362,880$, different orders that can yield nine factorial different inequalities. Coincidently, the same inequality would have resulted from lifting the variables in a different order for this example. Many times, a different lifting order will result in a different inequality, but frequently there are numerous repeated inequalities [5].

If different inequalities are created using different combinations of lifting, these inequalities may be combined to yield coefficients for each variable with an average value across

all the inequalities. For example, if x_1 had a coefficient of 2 in the first inequality and 3 in the second inequality, the average coefficient would be 2.5. Averaging the coefficients across all valid inequalities yields a new valid inequality. This inequality is not as strong as the inequalities developed through simultaneous lifting, as detailed in the following section.

2.3.5 Simultaneous Lifting

Simultaneously up lifting the variables of E results in inequalities of the form $\alpha \sum_{i \in E} w_i x_i + \sum_{i \in N \setminus E} \alpha_i x_i \leq \beta$, where $w_i \in R$ is a weight, as described by Gutierrez [21]. Clearly the goal is to seek the maximum α value for which this inequality is valid. Gutierrez also provided theory to show that the exact lifting coefficient can be obtained by solving a single integer program.

Zemel [44] calls his algorithm a simultaneous lifting algorithm, but more on this algorithm is discussed in the next section. Hooker and Easton [15] developed a linear time algorithm to simultaneously lift variables into cover inequalities for P_{KP}^{ch}. In 2009, Kubik expanded on this theory by creating a pseudopolynomial time algorithm that allows multiple simultaneously lifted sets to be sequentially lifted into a valid inequality for P_{KP}^{ch}.

2.3.6 Single vs. Synchronized Lifting

Further major types of lifting are single and synchronized lifting. To the best of the author's knowledge, this is the first written document to distinguish these types of lifting. Any application of single lifting generates exactly one inequality. Synchronized lifting generates multiple inequalities of the same form. The vast majority of methods are single lifting. Here

the focus is on synchronized lifting.

Zemel [44] developed the first exact method to simultaneously lift multiple variables in 1978, but this method required the use of exponentially many integer programs and can be applied only to cover inequalities from the binary Knapsack Problems. Zemel's method generates many inequalities by finding the extreme point of the polar, but is too computationally intensive to be efficiently implemented. In actuality Zemel's method generates numerous inequalities that are all simultaneously lifted. Thus, his method should have been classified as a synchronized simultaneous lifting algorithm.

Balas' well known result on the bounds of the lifting coefficients can now be viewed as a synchronized approximate up lifting technique [5]. Bolton [9] introduced synchronized simultaneous lifting (SSL), which is critical to this thesis and a brief description is provided here. Credit is given to Tommy Morrison for providing the synchronized name.

2.3.7 Synchronized Simultaneous Lifting

The input to SSL requires a knapsack constraint and two mutually exclusive lifting sets E_1 and E_2. SSL outputs valid inequalities of the form $\alpha_{E_1} \sum_{i \in E_1} x_i + \alpha_{E_2} \sum_{i \in E_2} x_i \leq 1$.

To begin SSL, the feasible combinations of the two sets are listed as ordered pairs. Next, beginning at the first extreme point on the axis, the slope of the lines from the first point to all other points is found. The line that does not eliminate any points is the most extreme. The slope of the line is computed through finding α values for each set. α_{E_1} is found by taking $(q - q^*)/(p^*q - q^*p)$, where p^* and q^* are the quantities from the first and second set, respectively, feasible at the first point, and p and q are the quantities from the first and

second set, respectively, feasible at the second point. The coefficient α_{E_2} is found by taking $(p^* - p)/(p^*q - q^*p)$.

The ratio of $\alpha_{E_2}/\alpha_{E_1}$ gives the slope of the line between the two points. By selecting the lowest ratio (or highest, depending on which axis is used as the starting point), the next extreme point can be found. Should a tie occur in the ratio of the α values, the point farthest down the list is selected. This corresponds to two or more points on the same line with the steepest slope. The point farthest out would be selected. This extreme point is now considered (p^*, q^*), and the slope of the lines to all subsequent points is found. This process is repeated until the final extreme point candidate located on the axis is selected as an extreme point.

Example 2.4 illustrates how SSL can be implemented in a KP to generate many valid inequalities. Reconsider the knapsack polytope from Example 2.1.

Example 2.4 $31x_1 + 30x_2 + 28x_3 + 25x_4 + 20x_5 + 20x_6 + 19x_7 + 18x_8 + 18x_9$

$$+17x_{10} + 17x_{11} + 15x_{12} + 14x_{13} + 14x_{14} + 13x_{15} + 12x_{16} \leq 94.$$

$$x_1, ..., x_{16} \in \{0, 1\}$$

For this example, arbitrarily set $E_1 = \{1, 2, 3, 4\}$ and $E_2 = \{5, 6, 7, 8, 9, 10,11,12,13,14,15,16\}$. As is seen above, the a coefficients have been sorted for each of the sets. All the feasible combinations of sets are found by starting the count for E_1 at its maximum possible and decreasing it while increasing E_2. The values listed in Table 2 are returned from the $FeasiblePoints$ subroutine. The right two columns are the amounts from each set that are feasible.

$count$	$feasE_1[count]$	$feasE_2[count]$
0	3	0
1	2	1
2	2	2
3	2	3
4	1	4
5	1	5
6	0	6

Table 2: Data Reported by $FeasiblePoints$ Subroutine

These are the potential candidates to be extreme points. A graphical representation of these points is shown in Figure 4.

By drawing the feasible points in a two dimensional coordinate, it is easy to see the relationship between the potential extreme point candidates. The slope of the line from the starting point at (3,0) to every other point is computed, and the steepest slope will be used. In this example, the next point chosen is (2,3), and so (2,3) becomes the next extreme point. This is also shown computationally in Table 3 below. The α_{E_1}, α_{E_2}, and $\alpha_{E_2}/\alpha_{E_1}$ values are given. Notice the ratio of the α values is lowest when the slope is the greatest.

E_1	E_2	α_{E_1}	α_{E_2}	$\frac{\alpha_{E_2}}{\alpha_{E_1}}$
2	1	$\frac{1}{3}$	$\frac{1}{3}$	1
2	2	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{2}$
2	**3**	$\mathbf{\frac{1}{3}}$	$\mathbf{\frac{1}{9}}$	$\mathbf{\frac{1}{3}}$
1	4	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{2}$
1	5	$\frac{1}{3}$	$\frac{2}{15}$	$\frac{2}{5}$
0	6	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{2}$

Table 3: Values for the first SSL inequality

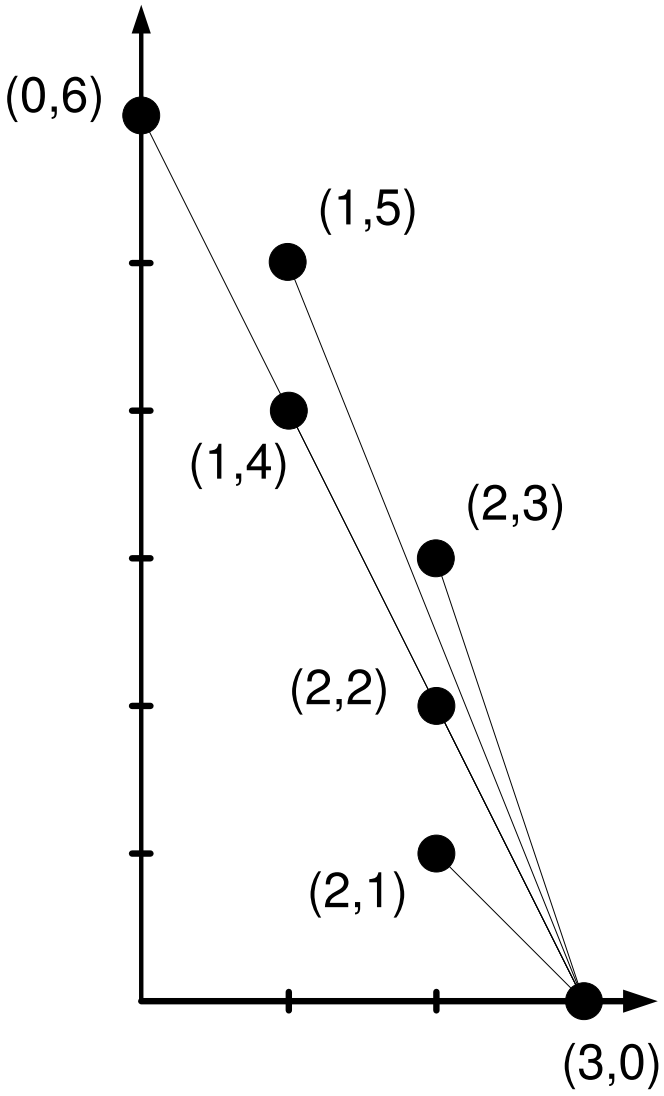

Figure 4: SSL algorithm finding next extreme point

The point $(2, 3)$ is chosen as the next extreme point, and the process is repeated for calculating the slopes to all subsequent points. Figure 5 illustrates the convex hull after all extreme points and α values have been found.

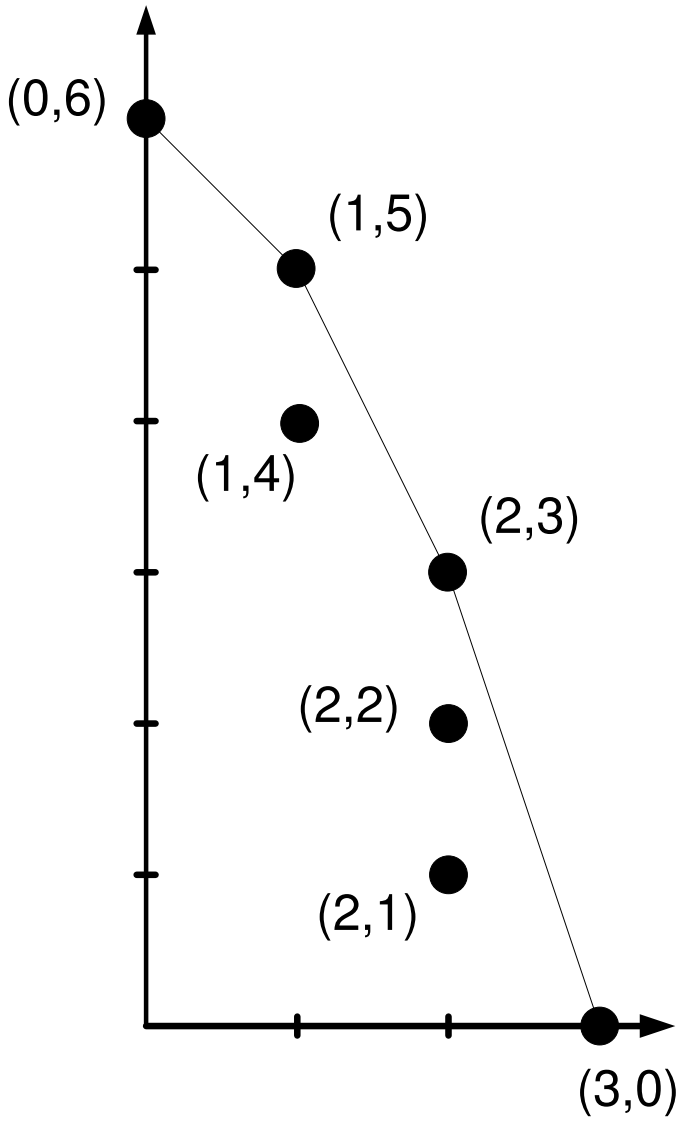

Figure 5: Final extreme points and inequalities produced

The valid inequalities produced in this example are $\frac{1}{3}\sum_{i\in E_1} x_i + \frac{1}{9}\sum_{i\in E_2} x_i \leq 1$, $\frac{1}{3}\sum_{i\in E_1} x_i + \frac{2}{15}\sum_{i\in E_2} x_i \leq 1$, and $\frac{1}{3}\sum_{i\in E_1} x_i + \frac{1}{6}\sum_{i\in E_2} x_i \leq 1$. The third inequality produced is facet defining, as shown in Figure 6 by the 16 affinely independent points.

0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	1	1	1	1	1	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	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	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	1	1	1	1	1	1
0	0	0	0	0	0	0	0	0	1	0	1	1	1	1	1
0	0	0	0	0	0	0	0	0	1	1	0	1	1	1	1
0	0	0	0	0	0	0	0	0	1	1	1	0	1	1	1
0	0	0	0	1	1	1	1	1	1	1	1	1	0	1	1
0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	1
0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0

Figure 6: Affinely independent points for $\frac{1}{3}\sum_{i\in E_1} x_i + \frac{1}{6}\sum_{i\in E_2} x_i \leq 1$

The other inequalities are not facet defining in the original problem. The inequality $\frac{1}{3}\sum_{i\in E_1} x_i + \frac{1}{9}\sum_{i\in E_2} x_i \leq 1$ contains 10 affinely independent points, and the inequality $\frac{1}{3}\sum_{i\in E_1} x_i + \frac{2}{15}\sum_{i\in E_2} x_i \leq 1$ contains only five affinely independent points.

Given this exciting SSL method and the new facet defining inequalities found in two dimensions, a logical next step was to create a synchronized simultaneously lifted algorithm for three dimensions. The next chapter introduces the contribution of this thesis, an algorithm that synchronized simultaneously lifts in three sets. Bring on the CATS!

3 Cutting-plane Algorithm for Three Sets

This chapter formally introduces the Cutting-plane Algorithm for Three Sets (CATS), which extends synchronized simultaneous lifting to three sets. The chapter begins by describing the inputs and outputs of CATS, followed with explanations of CATS' notation and the procedure for finding valid inequalities. CATS pseudocode and related proofs are shown. Finally, a full example that demonstrates how CATS can find previously undiscovered valid inequalities ends the chapter.

3.1 CATS Definitions and Steps

From a high level CATS requires two types of inputs: a knapsack constraint $\Sigma_{j \in N} a_j x_j \leq b$, and sets $E_1 = \{i_1^1, ..., i_{e_1}^1\}$, $E_2 = \{i_1^2, ..., i_{e_2}^2\}$ and $E_3 = \{i_1^3, ..., i_{e_3}^3\} \subset N$ such that $E_1,\ E_2,\ E_3 \neq \emptyset$, $E_1 \cap E_2 = \emptyset$, $E_1 \cap E_3 = \emptyset$, and $E_2 \cap E_3 = \emptyset$, where $e_1 = |E_1|$, $e_2 = |E_2|$ and $e_3 = |E_3|$. Without loss of generality, these sets are sorted in ascending order of their indices (descending order of their a coefficients).

CATS identifies the candidate extreme points and finds the convex hull of these extreme points. This convex hull, described now as P^{ch^3}, is a three dimensional polyhedron. Figure 7 depicts the extreme points (labeled 1-9) and the facets (A-G) of one such possible convex hull. The output of CATS are inequalities that form facets of this convex hull. These facets of P^{ch^3} take the form $\alpha_{E_1} y_1 + \alpha_{E_2} y_2 + \alpha_{E_3} y_3 \leq 1$. Consequently, the valid inequalities of P_{KP}^{ch} take the form $\alpha_{E_1} \sum_{i \in E_1} x_i + \alpha_{E_2} \sum_{i \in E_2} x_i + \alpha_{E_3} \sum_{i \in E_3} x_i \leq 1$. Clearly, the $y_i \geq 0$ and $x_i \geq 0$ inequalities are ignored as trivial facet defining inequalities of their respective polyhedrons.

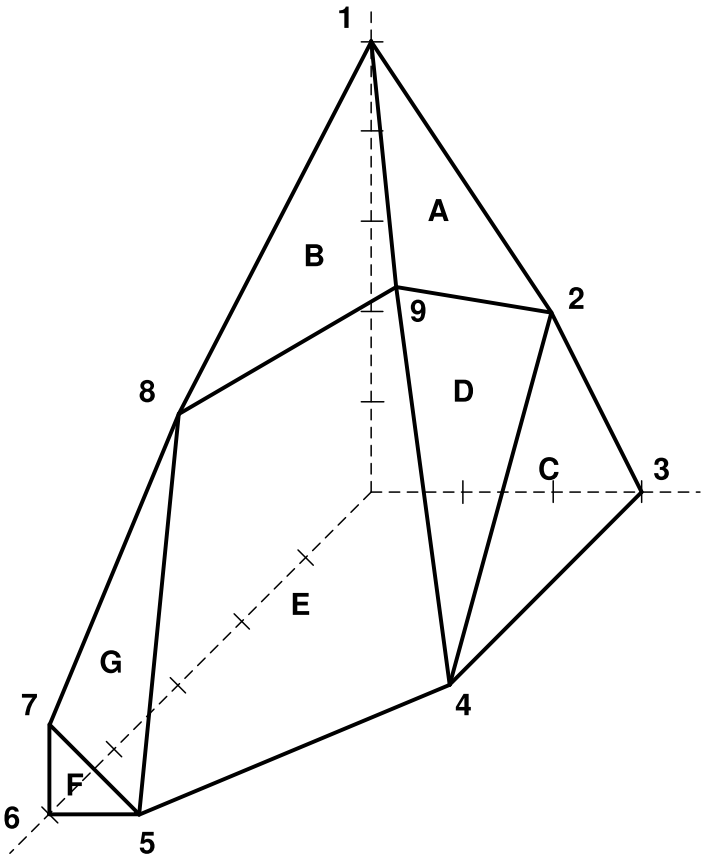

Figure 7: Example of a convex hull, P^{ch^3}, with facets in three dimensions

3.1.1 Starting CATS

Formally, CATS begins by identifying $EP = \{p^1, ..., p^{q'}\}$ as the set of potential candidates for extreme points around the convex hull. For $i = 1, ..., q'$ each $p^i = (p_1^i, p_2^i, p_3^i)$ represents a feasible point in P^{ch^3} and thus $\sum_{j=|E_1|-p_1^i+1}^{|E_1|} a_j + \sum_{j=|E_2|-p_2^i+1}^{|E_2|} a_j + \sum_{j=|E_3|-p_3^i+1}^{|E_3|} a_j \leq b$. Fortunately, these can be determined with $O(n^2)$ effort following Kubik's algorithm [27].

A critical component in CATS identifies adjacent points. These adjacent points create a Eulerian tour [16]. Every $p^i \in EP$ has an even number of adjacencies. These adjacencies are denoted as a left and right and the reader can create duplicate points to handle more than a single visit to a point in the extreme Eulerian tour. So define $adj_l^{p^i}$ and $adj_r^{p^i}$ to be the extreme point located to left and right of p^i for each $i = 1, ..., q'$. Every point in EP has a set of left and right adjacencies or no adjacent points. The left and right references are identified by envisioning oneself standing on that point looking into P^{ch^3}.

Next, CATS identifies the sets $EP_1 = \{p^i \in EP : p_1^i = 0\}$, $EP_2 = \{p^i \in EP : p_2^i = 0\}$ and $EP_3 = \{p^i \in EP : p_3^i = 0\}$. A modified version of Bolton's SSL algorithm [9] is used independently on EP_1, EP_2 and EP_3. This modified version identifies the actual extreme points on the respective E_1, E_2, and E_3 axes and identifies the adjacent extreme points for these points in the obvious fashion.

CATS now begins the main step. A candidate extreme point p^i with a nonempty $adj_l^{p^i}$ is identified and labeled as the middle point and called p^m. The analogy for searching for a violated point can be made of a windshield wiper sweeping across a windshield. Figure 8 shows the iterative process for finding the valid cutting planes.

The two adjacent points to p^m become $p^r = adj_r^{p^i}$ and $p^l = adj_l^{p^i}$. A plane is created that passes through the three points p^m, p^l, and p^r. Clearly, the plane, takes the form $\alpha_{E_1} y_1 + \alpha_{E_2} y_2 + \alpha_{E_3} y_3 = 1$. The formula to calculate the equation of this plane is obtained by solving the following system of equations for α_{E_1}, α_{E_2} and α_{E_3}.

$$\alpha_{E_1} p_1^m + \alpha_{E_2} p_2^m + \alpha_{E_3} p_3^m = 1$$

$$\alpha_{E_1} p_1^l + \alpha_{E_2} p_2^l + \alpha_{E_3} p_3^l = 1$$

$$\alpha_{E_1} p_1^r + \alpha_{E_2} p_2^r + \alpha_{E_3} p_3^r = 1.$$

The solution to this system of equations is

$$d = (p_1^m(p_2^l p_3^r - p_2^r p_3^l) + p_1^l(p_2^r p_3^m - p_2^m p_3^r) + p_1^r(p_2^m p_3^l - p_2^l p_3^m)). \qquad (i)$$

$$\alpha_{E_1} = (p_2^m(p_3^l - p_3^r) + p_2^l(p_3^r - p_3^m) + p_2^r(p_3^m - p_3^l))/d.$$

$$\alpha_{E_2} = (p_3^m(p_1^l - p_1^r) + p_3^l(p_1^r - p_1^m) + p_3^r(p_1^m - p_1^l))/d.$$

$$\alpha_{E_3} = (p_1^m(p_2^l - p_2^r) + p_1^l(p_2^r - p_2^m) + p_1^r(p_2^m - p_2^l))/d.$$

Thus, $\alpha_{E_1} y_1 + \alpha_{E_2} y_2 + \alpha_{E_3} y_3 \leq 1$ is a candidate cutting plane of P^{ch^3}. CATS calls a subroutine to check whether or not this inequality is valid.

3.1.2 Valid Inequality Check

This valid inequality subroutine verifies that all points $p^i \in EP$ satisfy this inequality by testing whether or not $\alpha_{E_1} p_1^i + \alpha_{E_2} p_2^i + \alpha_{E_3} p_3^i \leq 1$. Every time a p^i is found such that $\alpha_{E_1} p_1^i + \alpha_{E_2} p_2^i + \alpha_{E_3} p_3^i > 1$, as depicted Figure 8, then the candidate inequality is not valid. In this situation a new plane is created from the points p^m, p^l and p^i by solving the obvious system of equations (i). This new inequality is verified by continuing to check the remaining

points in EP. Once all points have been tested, the inequality is valid, stored as T^t, t is increased by 1 and a subroutine to find all adjacent extreme points in this plane is called.

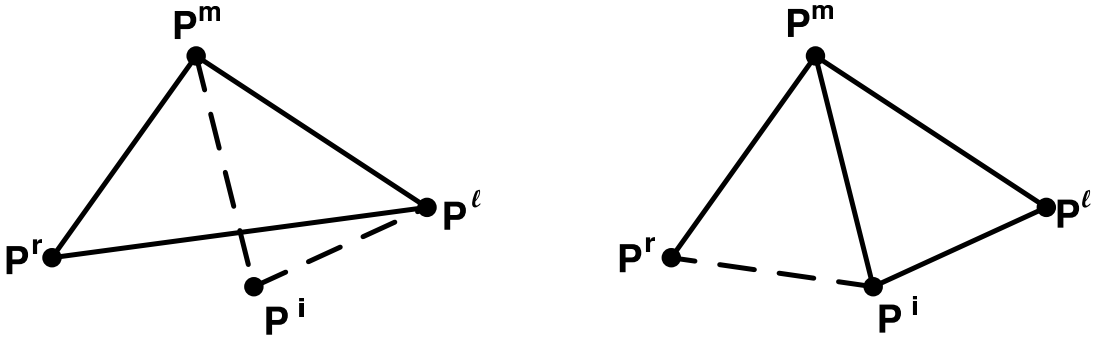

Figure 8: Creating new planes from a violated point

3.1.3 Extreme Plane Routine

The extreme plane routine begins by finding all the extreme points in EP that meet the valid inequality at equality. So $EE = \{p^i \in EP : \alpha_{E_1} p_1^i + \alpha_{E_2} p_2^i + \alpha_{E_3} p_3^i = 1\}$. To find the extreme points, begin by setting $q^r := p^m$ and $q^m := p^l$.

The following procedure determines which $p^i \in EE \setminus \{p^l\}$ is adjacent to p^l and thus it is done for each $p^i \in EE \setminus \{p^l\}$. Calculate the plane passing through p^l, p^i and the point (r', r', r') where r' is an irrational number larger than 0 and less than $\frac{1}{3}$. This can be achieved by solving for three temporary alpha values, α'_1, α'_2 and α'_3, similar as before with a right hand side of 1. Clearly, this plane must exist since r' is irrational and both p^l and p^i are rational.

If either every $p^j \in EE$ has $\alpha'_1 p_1^j + \alpha'_2 p_2^j + \alpha'_3 p_3^j \leq 1$ and $\alpha'_1 q_1^r + \alpha'_2 q_2^r + \alpha'_3 q_3^r < 1$ or every such p^j has $\alpha'_1 p_1^j + \alpha'_2 p_2^j + \alpha'_3 p_3^j \geq 1$ and $\alpha'_1 q_1^r + \alpha'_2 q_2^r + \alpha'_3 q_3^r > 1$, then p^i is a candidate adjacent point to the right of $p^l = q^m$. Once all $p^i \in EE$ have been tested, let p^k be the

point that is a candidate adjacent point to the right of p^l and is farthest from p^l. Therefore, p^k is the adjacent point to the right of p^l, so assign $adj_r^{p^l} := p^k$, $adj_l^{p^k} := p^l$, $adj_l^{p^m} := p^k$ and $adj_r^{p^k} := p^m$. Now let $q^r := q^m$ and $q^m := p^k$, and now continue this process in the obvious fashion by creating and testing the planes containing q^m, p^i and (r', r', r') for each $i \in EE$. This procedure terminates once q^m is assigned to p^m. At this point assign $adj_r^{q^l} := p^r$ and $adj_l^{p^r} := q^l$. Observe that this *ExtremePlaneRoutine* replaces the edge p^m to $adj_l^{p^m}$ in the Eulerian tour with the path of the adjacent extreme points proceeding from $adj_l^{p^m}$ to p^m. A graphical explanation of this instance is given in the next section and in Figure 11.

Figure 9 describes the extreme point routine graphically. Notice that all points except for (r', r', r') are in the plane created by p^l, p^m and p^r. The point p^A is found as the adjacent point to p^l. Notice also how in the second graphic the temporary plane bissects the points in EE, with p^A on one side and the other points are on the opposite. Thus, the point p^B is not adjacent to p^l.

CATS continues to seek valid inequalities stemming from p^m until the following condition is met: $adj_r^{p^m} = p^i$, which is also $adj_r^{p^m} = adj_l^{p^m}$. Alternatively, when an inequality passing through the original adjacent point right to p^m is found to be valid, all valid inequalities from p^m have been exhausted. Referring back to the windshield wiper analogy, this is essentially the wiper hitting its terminal point, p^r, after rotating across the width of the windshield.

3.1.4 The Eulerian Tour and Termination

Now that all valid inequalities are found from p^m, it can be removed and a new point is chosen to repeat this process. So let $EP := EP \setminus \{p^m\}$. Let $p^j \in EP$ such that $adj_l^{p^j} \neq \emptyset$,

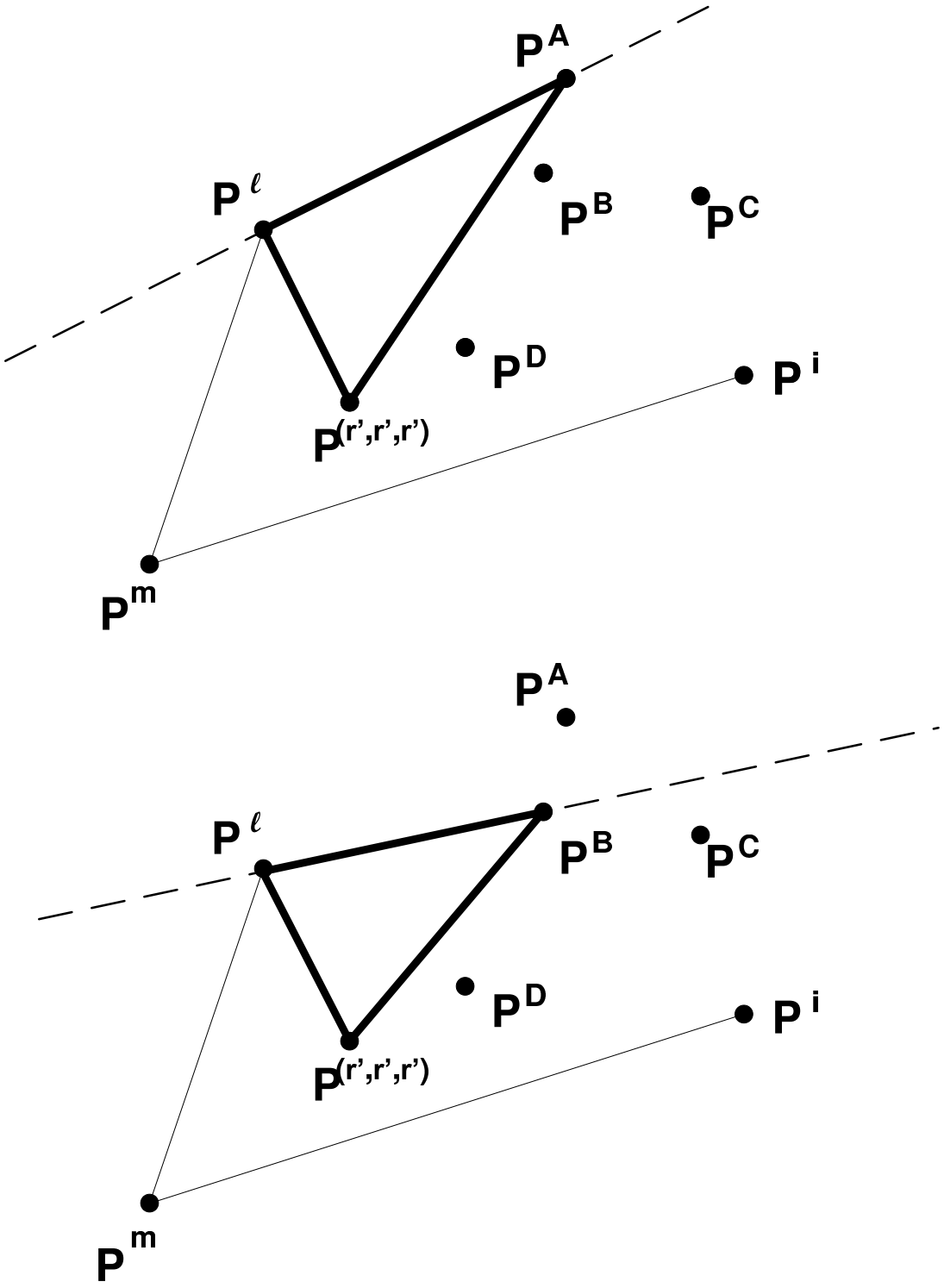

Figure 9: Example of the extreme plane determining next adjacent point to p^l

if none exists, then the algorithm terminates. Else let $p^m := p^j,\ p^l := adj_l^{p^j}$ and $p^r := adj_r^{p^j}$. This entire process repeats until the termination condition is satisfied and then the algorithm terminates by reporting all valid inequalities of P^{ch^3}.

It could now be inferred by the reader that the adjacency list inducing the extreme Eulerian tour is what enables CATS to define the convex hull. Initially, all the points on the axes have adjacent points. As new inequalities are defined by points located within the convex hull away from the axes, the adjacency lists must also be modified. If one would follow the adjacency list from one point to the next, it is possible to transverse a portion of the convex hull and arrive at the original point. This is because the adjacency lists from each extreme point define a Eulerian tour, and no matter what new points are found within the convex hull, this tour remains intact. However, through the updating process, any point with identical left and right adjacencies is removed from EP. Figures 10 and 11 illustrate the progression of the Eulerian tour throughout CATS.

In the two figures, notice how the heavy line begins on the convex hull axes, illustrating the adjacencies of the points located on the axes. As CATS develops valid cutting planes, the Eulerian tour follows the newly verified plane. As more planes are confirmed, the tour becomes smaller until, and as seen on the lower right graphic of Figure 11, the Eulerian tour contains only one plane. At this point, all facets in P^{ch^3} have been identified. CATS will continue until termination, but no more inequalities are found. Notice points with $adj_r = adj_l$ are illustrated with a dashed line and removed from EP and the Eulerian tour.

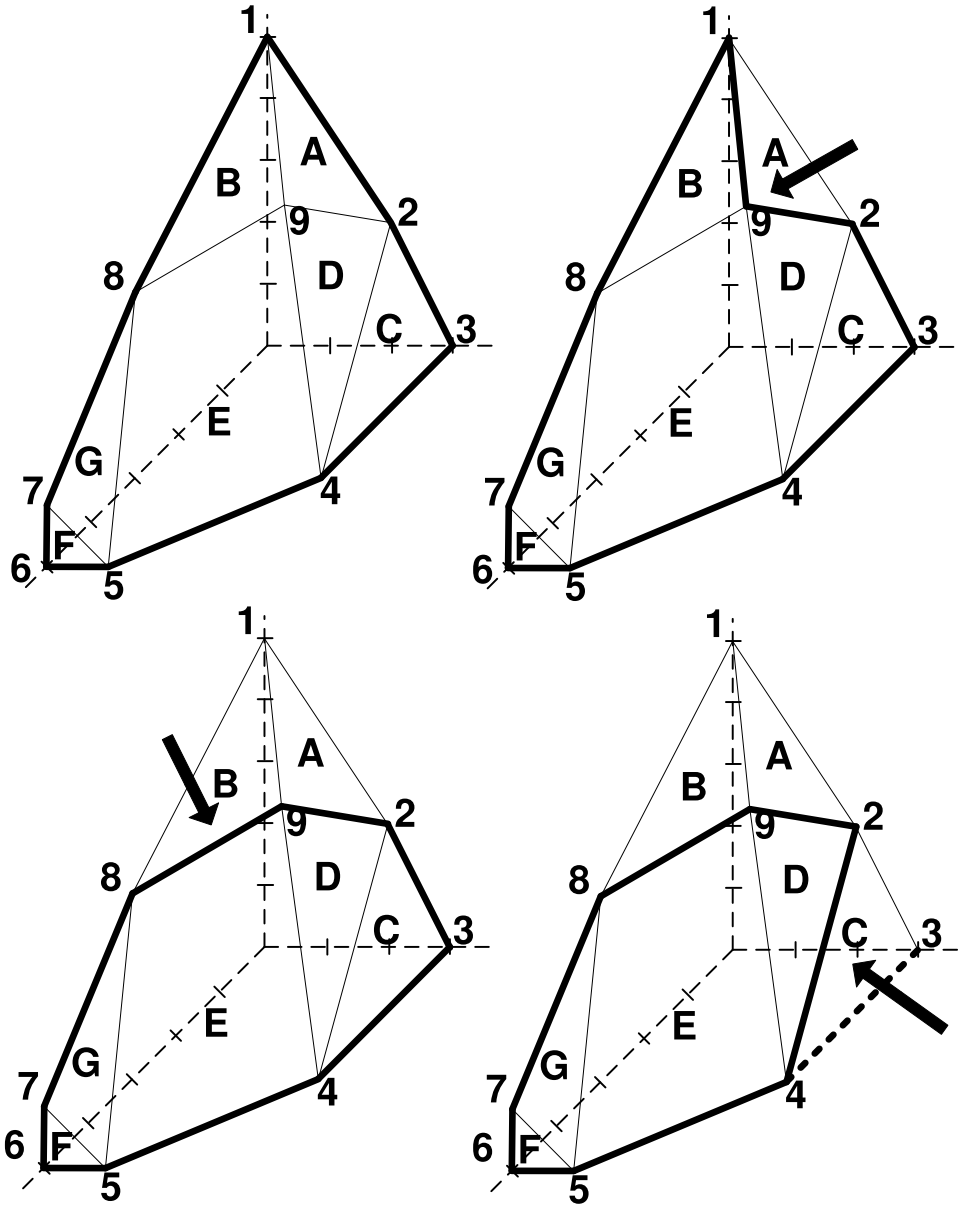

Figure 10: Evolution of Eulerian tour

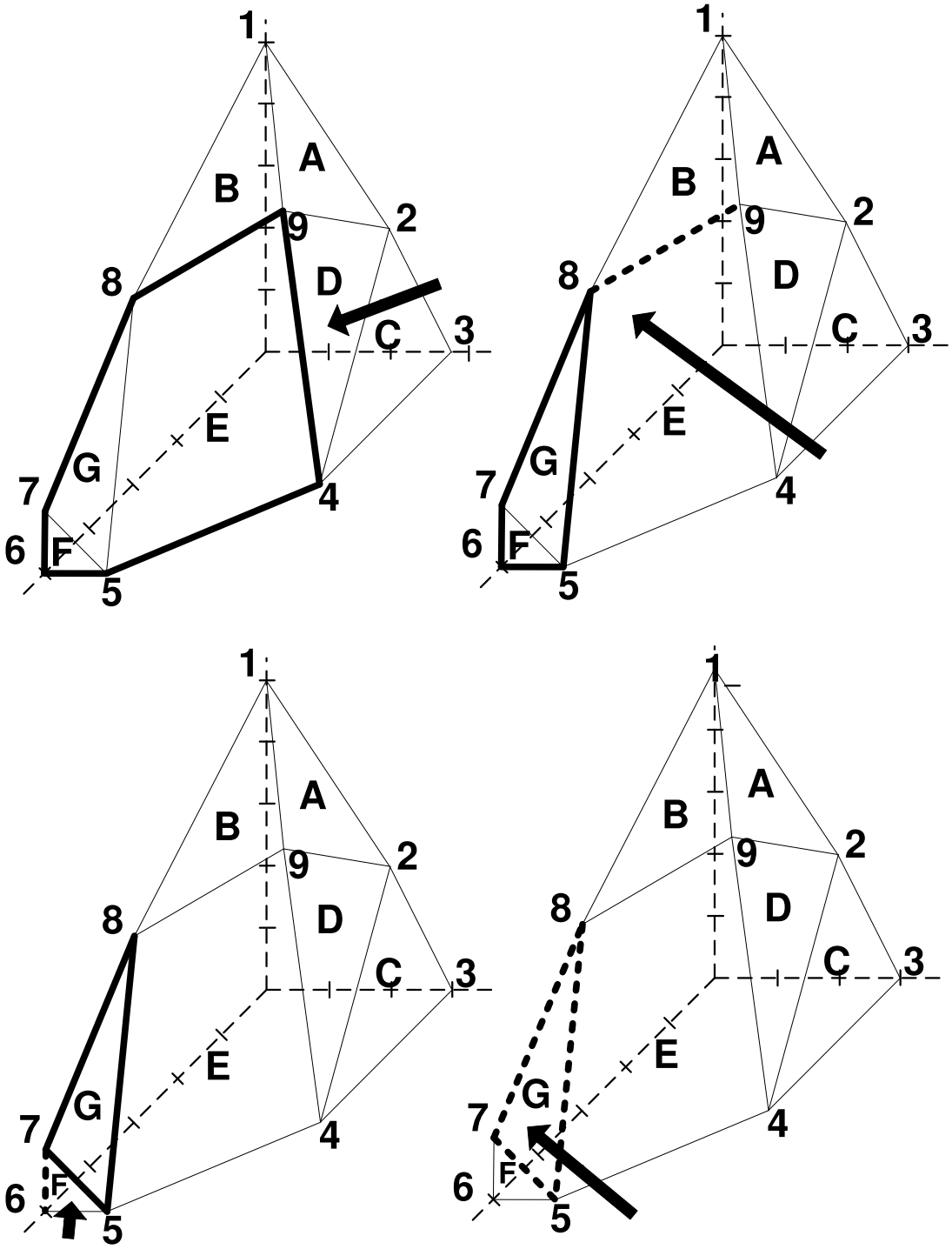

Figure 11: Evolution of Eulerian tour continued

3.2 Pseudocode for CATS

The pseudocode for CATS includes the main step of CATS, the Extreme Plane Adjacencies routine, and modified versions of Kubik's feasible point subroutine and Bolton's algorithm to find adjacencies. Formally, the pseudocode for CATS is as follows.

CATS

Initialization

Use Kubik's Feasible Point Subroutine in Three Sets to define candidate extreme points $EP := \{p^1, ..., p^{q'}\}$.

Use Bolton's SSL Subroutine on each of the three axises, EP_1, EP_2 and EP_3, to define the left and right adjacencies for some of the points in EP.

Main Step

While there exists a p^i with $adj_l^{p^i} \neq \emptyset$ and $adj_l^{p^i} \neq adj_r^{p^i}$ *begin*

Check the adjacency lists and remove any $p^{j'}$ that has $adj_r^{p^{j'}} = adj_l^{p^{j'}}$ from EP.

Let $p^m := p^i$, $orig^r := adj_r^{p^i}$ and $p^l := adj_l^{p^i}$.

While $adj_r^{p^l} \neq orig_r$ *begin*

$p^r := orig^r$.

Find α_1, α_2, α_3 on p^m, p^r and p^l using (i).

For $j = 1$ to q' *begin*

If $\alpha_1 p_1^j + \alpha_2 p_2^j + \alpha_3 p_3^j > 1$, *then begin*

$adj_r^{p^l} := p_j$, $adj_l^{p^j} := p_l$, $adj_l^{p^m} := p_j$ and $adj_r^{p^j} := p_m$.

$p^r := p_j$.

Find new α_1, α_2, α_3 on p^m, p^r and p^l using (i).

End if

End for

Call *Extreme Plane Adjacencies* to update the adjacencies.

Report $\alpha_1 \sum_{k \in E_1} x_k + \alpha_2 \sum_{k \in E_2} x_k + \alpha_3 \sum_{k \in E_3} x_k \leq 1$ as a valid inequality.

End while

$EP := EP \setminus \{p^m\}$.

End while

Extreme Plane Adjacencies

Let $EE = \{p^j \in EP : \alpha_1 p_1^j + \alpha_2 p_2^j + \alpha_3 p_3^j = 1\}$.

Let $q^m := p^l$ and $q^r := p^m$.

While $q^m \neq p^m$ *begin*

For each $p^i \in EE \setminus \{q^r, q^m\}$ *begin*

Use equation (i) to find α'_{E_1}, α'_{E_2} and α'_{E_3} on the points q^m, p^i and (r', r', r')

where r' is an irrational number $0 < r' < \frac{1}{3}$.

$CP := \emptyset$.

If all $p^j \in EE$ satisfy $\alpha'_1 p_1^j + \alpha'_2 p_2^j + \alpha'_3 p_3^j \leq 1$ and $\alpha'_1 q_1^r + \alpha'_2 q_2^r + \alpha'_3 q_3^r < 1$ or

all $p^j \in EE$ satisfy $\alpha'_1 p_1^j + \alpha'_2 p_2^j + \alpha'_3 p_3^j \geq 1$ and $\alpha'_1 q_1^r + \alpha'_2 q_2^r + \alpha'_3 q_3^r > 1$

then begin $CP := CP \cup \{p^j\}$, $p^k := p^j$.

If $adj_l^{p^k} \neq \emptyset$ *then begin*

$EP := EP \cup \{p^{k'}\}$ where $p^{k'}$ is a duplication of p^k.

$adj_l^{p^{k'}} := adj_l^{p^k}$, $adj_r^{p^{k'}} := adj_r^{p^k}$, $adj_r^{adj_l^{p^k}} := p^{k'}$ and $adj_l^{adj_r^{p^k}} := p^{k'}$.

End if

Let $adj_l^{p^k} := q^m$ and $adj_r^{q^m} := p^k$.

Let $q^r := q^m$ and $q^m := p^k$.

End For

End While

Let $adj_r^{q^r} := orig^r$ and $adj_l^{orig^r} := q^r$.

Check the adjacency lists and remove any $p^{j'}$ that has $adj_r^{p^{j'}} = adj_l^{p^{j'}}$ from EP.

Modified Three Set Feasible Point Algorithm [27]

For $p = 0$ *to* $|E_1|$

$q := 0$ and $sum := \Sigma_{k=1}^{p} a_{i^1_{e_1-k+1}}$.

While $sum \leq b$ *and* $q \leq |E_2|$ *begin*

$EP := EP \cup \{p^j\}$ where $p^j = (p, q, 0)$.

$q := q + 1$.

$sum := sum + a_{i^2_{e_2-q+1}}$.

End While

If $sum > b$, *then begin*

$sum := sum - a_{i^2_{e_2-q+1}}$ and $q := q - 1$.

While $q \geq 0$ *and* $r \leq |E_3|$ *begin*

If $sum \leq b$, *then begin*

$EP := EP \cup \{p^j\}$ where $p^j = (p, q, r)$.

$r := r + 1$ and $sum := sum + a_{i^3_{e_3-r+1}}$.

else

$sum := sum - a_{i^2_{e_2-q+1}}$ and $q := q - 1$.

End While

End If

End For

Modified Bolton's Algorithm to Find Adjacencies for E^3 [9]

Initialization:

Let $EP_3 = \{e^1, ..., e^f\}$ be the reduced set of points in reverse lexicographic order on

the E^3 axis in EP. Let $e^i = (p^i, q^i, 0)$ for $i = 1, ..., f$. Note the cases for EP_2 and EP_1 follow similar logic.

Let $p^* = max\ p^i \in EP_3 : q^i = 0$ where i^* is the argument where p^* occurs and $q^* = 0$.

While $p^* > 0$ *begin*

$count =: i^* + 1$ and $\alpha := \infty$.

While count $\leq f$ *begin*

$p := p^{count}$ and $q := q^{count}$.

$\alpha' := \frac{q - q^*}{p^* q - q^* p}$ and $\alpha'' := \frac{p^* - p}{p^* q - q^* p}$.

If $\alpha' \neq 0$, *then begin*

If $\alpha \geq \frac{\alpha''}{\alpha'}$, *then begin*

$\alpha := \frac{\alpha''}{\alpha'}$, $mark := count$.

$count := count + 1$.

End While

$adj_l^{e^{i^*}} := e^{mark}$, $adj_r^{e^{mark}} := e^{i^*}$.

$p^* := p^{mark}$ and $q^* := q^{mark}$.

End While

3.3 CATS' Theoretical Results

The following section describes the valid inequalities produced by CATS, as well as proofs of termination, runtime, and capability of producing facet defining inequalities. The termination of CATS is dependent upon the Eulerian tour, which decreases the number of nodes every time a p^m is removed from EP. CATS runtime is due to the steps through which CATS finds and confirms valid inequalities. The final proof describes the conditions for facet defining in P_{KP}^{ch}.

Lemma 3.1 *Every inequality reported by CATS is valid for* P_{KP}^{ch}.

Proof: Let $\alpha_{E_1} \sum_{i \in E_1} x_i + \alpha_{E_2} \sum_{i \in E_2} x_i + \alpha_{E_3} \sum_{i \in E_3} x_i \leq 1$ be any inequality reported from CATS. Kubik's feasible point routine identifies all potential candidates for extreme points called EP. CATS only reports an inequality if every $p^i \in EP$ satisfies the inequality. Thus, $\alpha_{E_1} y_1 + \alpha_{E_2} y_2 + \alpha_{E_3} y_3 \leq 1$ is a valid inequality of P^{ch^3}. Since Kubik's algorithm uses the sorted order of the knapsack constraint, no points in P_{KP}^{ch} violate $\alpha_{E_1} \sum_{i \in E_1} x_i + \alpha_{E_2} \sum_{i \in E_2} x_i + \alpha_{E_3} \sum_{i \in E_3} x_i \leq 1$ and so it is valid.

□

Theorem 3.2 *CATS terminates in a finite number of steps and reports all nontrivial facet defining inequalities of* P^{ch^3}.

Proof: Clearly, the initialization runs in a finite number of steps and correctly identifies both the set of potential extreme points and the set of adjacent extreme points along the E_1, E_2 and E_3 axes due to Kubik's and Bolton's theorems. The remainder will be shown

algorithmically.

The main step begins with a Eulerian tour of adjacent extreme points and selects one such extreme point p^m. Furthermore, p^m, $adj_l^{p^m}$ and $adj_r^{p^m}$ are all known to be extreme points. CATS finds a facet defining inequality that contains p^m and $adj_l^{p^m}$ with a point that is a candidate p^j extreme point. Now it must be shown that the next iteration maintains a Eulerian tour of adjacent extreme points.

CATS finds all points on the plane between p^m, $adj_l^{p^m}$ and p^j, which form the set EE. CATS identifies the extreme adjacent points in EE by creating a plane between the irrational point (r', r', r'), the most recently known extreme point ($adj_l^{p^m}$ in the first case) and some other point p^k in EE. The point (r', r', r') is guaranteed to be in the interior of P^{ch^3}. If this plane has all points in EE on one side and the second most recent extreme point (p^m in the first case) is not on this plane, then p^k is a candidate for an extreme point. From all of these candidate extreme points, the one that is farthest from the most recent extreme point is selected as the next extreme point. The adjacencies are updated and the process continues until p^m is reached.

Once p^m is reached the most recent extreme point becomes the left adjacent of p^m, which has successfully updated the Eulerian tour. If the left adjacent of p^m is not p^r, then there still remains a Eulerian tour of adjacent extreme points. If not, then p^m is removed from EP and all facet defining inequalities of P^{ch^3} that abut p^m between $adj_l^{p^m}$ and $adj_r^{p^m}$ are found.

Once p^m is removed, there still remains a Eulerian tour of adjacent extreme points. So this process continues. Since there is a finite set of points in EP, this process terminates in a finite number of steps and furthermore CATS reports all nontrivial facet defining inequalities

of P^{ch^3}.

□

Theorem 3.3 *CATS requires $O(n^4|T|)$ effort where $|T|$ is the number of generated inequalities.*

Proof: Kubik showed that her subroutine runs in $O(n^2)$. Furthermore, this algorithm generates at most $O(n^2)$ points. Thus, $|EP|$ is on the order of $O(n^2)$. Bolton proved that her algorithm runs in $O(n^2)$ and the modified version is a subset of this algorithm and so it also runs in at most $O(n^2)$ effort.

The main step of CATS runs in $O(n^4|T|)$. In each iteration, this step finds a single inequality. Furthermore, all of the points in EP are examined exactly one time to create this inequality, thus determining a single valid inequality requires $O(n^2)$.

Once this plane is determined, *extreme plane* is called and requires $O(n^4)$ effort. To see this $|EE|$ is bounded by $O(n^2)$. Furthermore to determine the next adjacent point requires scanning through all points in EE at most n^2 times. There can be at most n^2 such adjacencies and this subroutine is therefore bounded by $O(n^4)$.

Therefore, the main step of CATS requires $O(n^4|T|)$ effort where $|T|$ is the number of generated inequalities.

□

The class of valid inequalities produced by CATS is capable of being facet defining. To test if these inequalities are facet defining, use the conditions outlined in the following

theorem.

Theorem 3.4 *Let* $\alpha_{E_1}\sum_{i\in E_1}x_i+\alpha_{E_2}\sum_{i\in E_2}x_i+\alpha_{E_3}\sum_{i\in E_3}x_i\leq 1$ *be an inequality reported from CATS. Furthermore, let this inequality be generated from the points* $p^{t^1}=(p_1^{t^1},p_2^{t^1},p_3^{t^1})$, $p^{t^2}=(p_1^{t^2},p_2^{t^2},p_3^{t^2})$ *and* $p^{t^3}=(p_1^{t^3},p_2^{t^3},p_3^{t^3})$. *If there exists an ordering of these three points that meets the following four conditions*

i) $1\leq p_1^{t^1}\leq e_1-1$ *and*

a) if $p_1^{t^1}=1$, *then the set* $\{i_1^1,i^2_{e_2-p_2^{t^1}+1},...,i^2_{e_2},i^3_{e_3-p_3^{t^1}+1},...,i^3_{e_3}\}$ *is not a cover;*

b) if $2\leq p_1^{t^1}\leq e_1-2$, *then the sets* $\{i_1^1,i^1_{e_1-p_1^{t^1}+2},...,i^1_{e_1},i^2_{e_2-p_2^{t^1}+1},...,i^2_{e_2},i^3_{e_3-p_3^{t^1}+1},...,i^3_{e_3}\}$ *and* $\{i^1_{e_1-p_1^{t^1}},...,i^1_{e_1-1},i^2_{e_2-p_2^{t^1}+1},...,i^2_{e_2},i^3_{e_3-p_3^{t^1}+1},...,i^3_{e_3}\}$ *are not covers.*

c) if $p_1^{t^1}=e_1-1$, *then the set* $\{i_1^1,...,i^1_{e_1-1},i^2_{e_2-p_2^{t^1}+1},...,i^2_{e_2},i^3_{e_3-p_3^{t^1}+1},...,i^3_{e_3}\}$ *is not a cover.*

ii) $1\leq p_2^{t^2}\leq e_2-1$ *and*

a) if $p_2^{t^2}=1$, *then the set* $\{i^1_{e_1-p_1^{t^2}+1},...,i^1_{e_1},i_1^2,i^3_{e_3-p_3^{t^2}+1},...,i^3_{e_3}\}$ *is not a cover;*

b) if $2\leq p_2^{t^2}\leq e_2-2$, *then the sets* $\{i^1_{e_1-p_1^{t^2}+1},...,i^1_{e_1},i_1^2,i^2_{e_2-p_2^{t^2}+2},...,i^2_{e_2},i^3_{e_3-p_3^{t^2}+1},...,i^3_{e_3}\}$ *and* $\{i^1_{e_1-p_1^{t^2}+1},...,i^1_{e_1},i^2_{e_2-p_2^{t^2}},...,i^2_{e_2-1},i^3_{e_3-p_3^{t^2}+1},...,i^3_{e_3}\}$ *are not covers.*

c) if $p_2^{t^2}=e_2-1$, *then the set* $\{i^1_{e_1-p_1^{t^2}+1},...,i^1_{e_1},i_1^2,...,i^2_{e_2-1},i^3_{e_3-p_3^{t^2}+1},...,i^3_{e_3}\}$ *is not a cover.*

iii) $1\leq p_3^{t^3}\leq e_3-1$ *and*

a) if $p_2^{t^3}=1$, *then the set* $\{i^1_{e_1-p_1^{t^3}+1},...,i^1_{e_1},i^2_{e_2-p_2^{t^3}+1},...,i^2_{e_2},i_1^3\}$ *is not a cover;*

b) if $2 \leq p_3^{t^3} \leq e_3 - 2$, *then the sets* $\{i^1_{e_1-p_1^{t^3}+1}, ..., i^1_{e_1}, i^2_{e_2-p_2^{t^3}+1}, ..., i^2_{e_2}, i^3_1, i^3_{e_3-p_3^{t^3}+2}, ..., i^3_{e_3}\}$

and $\{i^1_{e_1-p_1^{t^3}+1}, ..., i^1_{e_1}, i^2_{e_2-p_2^{t^3}}, ..., i^2_{e_2-1}, i^3_{e_3-p_3^{t^3}}, ..., i^3_{e_3-1}\}$ *are not covers.*

c) if $p_3^{t^3} = e_3 - 1$, *then the set* $\{i^1_{e_1-p_1^{t^3}+1}, ..., i^1_{e_1}, i^2_{e_2-p_2^{t^3}+1}, ..., i^2_{e_2}, i^3_1, ..., i^3_{e_3-1}\}$

is not a cover.

iv) At least one of the following three sets is not a cover where j *is the smallest index in*

$N \setminus (E_1 \cup E_2 \cup E_3)$

a) $\{j, i^1_{e_1-p_1^{t^1}+1}, ..., i^1_{e_1}, i^2_{e_2-p_2^{t^1}+1}, ..., i^2_{e_2}, i^3_{e_3-p_3^{t^1}+1}, ..., i^3_{e_3}\}$,

b) $\{j, i^1_{e_1-p_1^{t^2}+1}, ..., i^1_{e_1}, i^2_{e_2-p_2^{t^2}+1}, ..., i^2_{e_2}, i^3_{e_3-p_3^{t^2}+1}, ..., i^3_{e_3}\}$, *or*

c) $\{j, i^1_{e_1-p_1^{t^3}+1}, ..., i^1_{e_1}, i^2_{e_2-p_2^{t^3}+1}, ..., i^2_{e_2}, i^3_{e_3-p_3^{t^3}+1}, ..., i^3_{e_3}\}$.

Then $\alpha_{E_1} \sum_{i \in E_1} x_i + \alpha_{E_2} \sum_{i \in E_2} x_i + \alpha_{E_3} \sum_{i \in E_3} x_i \leq 1$ *is facet defining for* P^{ch}_{KP}.

Proof: Let $\alpha_{E_1} \sum_{i \in E_1} x_i + \alpha_{E_2} \sum_{i \in E_2} x_i + \alpha_{E_3} \sum_{i \in E_3} x_i \leq 1$ be some inequality returned from CATS. This inequality is valid due to Lemma 3.1. Let F be the face of this valid inequality, $F = \{x \in P^{ch}_{KP} : \alpha_{E_1} \sum_{i \in E_1} x_i + \alpha_{E_2} \sum_{i \in E_2} x_i + \alpha_{E_3} \sum_{i \in E_3} x_i = 1\}$. Furthermore, the origin is not in F and thus $F \neq P^{ch}_{KP}$ so $dim(F) \leq n-1$. Now it suffices to find n affinely independent points in F. All x_i's are zeros except the x_i's mentioned specifically.

The first e_1 points follow the assumptions of ia), ib) or ic). If ia) is true, then the e_1 points are $x_i = 1$, if $i \in \{i^2_{e_2-p_2^{t^1}+1}, ..., i^2_{e_2}, i^3_{e_3-p_3^{t^1}+1}, ..., i^3_{e_3}\}$ and $x_j = 1$ for each $j \in E_1$. If ib) is true, then the e_1 points are $x_i = 1$ if $i \in \{i^1_{e_1-p_1^{t^1}+2}, ..., i^1_{e_1}, i^2_{e_2-p_2^{t^1}+1}, ..., i^2_{e_2}, i^3_{e_3-p_3^{t^1}+1}, ..., i^3_{e_3}\}$ and $x_j = 1$ for each $j \in \{i^1_1, ...i^1_{e_1-p_1^{t^1}-1}\}$. The remainder of the e_1 points for ib) occur when $x_i = 1$, if $i \in \{i^2_{e_2-p_2^{t^1}+1}, ..., i^2_{e_2}, i^3_{e_3-p_3^{t^1}+1}, ..., i^3_{e_3}\}$ and also $x_i = 1$ for all $i \in \{i^1_{e_1-p_1^{t^1}}, ..., i^1_{e_1}\} \setminus \{j\}$

for each $j \in \{i^1_{e_1-p^{t^1}_1}, ..., i^1_{e_1}\}$. Finally, if ic) is true, then $x_i = 1$ for all $i \in \{i^1_1, ..., i^1_{e_1}\} \setminus \{j\}$ for each $j \in \{i^1_1, ..., i^1_{e_1}\}$.

The next e_2 points follow the assumptions of iia), iib) or iic). If iia) is true, then the e_2 points are $x_i = 1$, if $i \in \{i^1_{e_1-p^{t^2}_1+1}, ..., i^1_{e_1}, i^3_{e_3-p^{t^2}_3+1}, ..., i^3_{e_3}\}$ and $x_j = 1$ for each $j \in E_2$. If iib) is true, then the e_2 points are $x_i = 1$ if $i \in \{i^1_{e_1-p^{t^2}_1+2}, ..., i^1_{e_1}, i^2_{e_2-p^{t^2}_2+1}, ..., i^2_{e_2}, i^3_{e_3-p^{t^2}_3+1}, ..., i^3_{e_3}\}$ and $x_j = 1$ for each $j \in \{i^2_1, ...i^2_{e_2-p^{t^2}_1-1}\}$. The remainder of the e_2 points for iib) occur $x_i = 1$, if $i \in \{i^1_{e_1-p^{t^2}_1+1}, ..., i^1_{e_1}, i^3_{e_3-p^{t^2}_3+1}, ..., i^3_{e_3}\}$ with $x_i = 1$ for all $\{i^2_{e_2-p^{t^2}_2}, ..., i^2_{e_2}\} \setminus \{j\}$ for each $j \in \{i^2_{e_2-p^{t^2}_2}, ..., i^2_{e_2}\}$. Finally, if iic) is true, then $x_i = 1$ for all $\{i^2_1, ..., i^2_{e_2}\} \setminus \{j\}$ for each $j \in \{i^2_1, ..., i^2_{e_2}\}$.

The next e_3 points follow the assumptions of iiia), iiib) or iiic). If iiia) is true, then the e_3 points are $x_i = 1$, if $i \in \{i^1_{e_1-p^{t^3}_1+1}, ..., i^1_{e_1}, i^2_{e_2-p^{t^3}_2+1}, ..., i^2_{e_2}\}$ and $x_j = 1$ for each $j \in E_3$. If iiib) is true, then the e_3 points are $x_i = 1$ if $i \in \{i^1_{e_1-p^{t^3}_1+2}, ..., i^1_{e_1}, i^2_{e_2-p^{t^3}_2+1}, ..., i^2_{e_2}, i^3_{e_3-p^{t^3}_3+1}, ..., i^3_{e_3}\}$ and $x_j = 1$ for each $j \in \{i^3_1, ...i^3_{e_3-p^{t^3}_1-1}\}$. The remainder of the e_3 points for iiib) occur $x_i = 1$, if $i \in \{i^1_{e_1-p^{t^3}_2+1}, ..., i^1_{e_1}, i^2_{e_2-p^{t^3}_3+1}, ..., i^2_{e_2}\}$ with $x_i = 1$ for all $\{i^3_{e_3-p^{t^3}_3}, ..., i^3_{e_3}\} \setminus \{j\}$ for each $j \in \{i^3_{e_3-p^{t^3}_3}, ..., i^3_{e_3}\}$. Finally, if iiic) is true, then $x_i = 1$ for all $\{i^3_1, ..., i^3_{e_3}\} \setminus \{j\}$ for each $j \in \{i^3_1, ..., i^3_{e_3}\}$.

The final set of points are generated from iv). If iva) is true, then these points are $x_i = 1$ for all $i \in \{i^1_{e_1-p^{t^1}_1+1}, ..., i^1_{e_1}, i^2_{e_2-p^{t^1}_2+1}, ..., i^2_{e_2}, i^3_{e_3-p^{t^1}_3+1}, ..., i^3_{e_3}\}$ and $x_j = 1$ for each $j \in N \setminus (E_1 \cup E_2 \cup E_3)$.

Consequently, there are n points that meet $\alpha_{E_1} \sum_{i \in E_1} x_i + \alpha_{E_2} \sum_{i \in E_2} x_i + \alpha_{E_3} \sum_{i \in E_3} x_i \leq 1$ at equality and so are in its face. Since p^{t^1}, p^{t^2} and p^{t^3} induce a plane in P^{ch^3}, they are affinely independent. Thus, these n points are also affinely independent. So the induced face has

dimension $n-1$ and it is facet defining.

□

Using the notation and procedures described previously, this section provides a detailed example of the algorithm and the inequalities produced in CATS. After reading a portion of this section, the reader may realize the figures from this example were used to illustrate the algorithm generally in the earlier section. The convex hull, illustrated to give the reader a sense of P^{ch^3}, is defined through this knapsack constraint. The reader may choose to reference these figures throughout this section.

3.4 CATS Example

Recall the knapsack constraint from Example 2.1:

$$31x_1 + 30x_2 + 28x_3 + 25x_4 + 20x_5 + 20x_6 + 19x_7 + 18x_8 + 18x_9 + 17x_{10} + 17x_{11} + 15x_{12} +$$

$$14x_{13} + 14x_{14} + 13x_{15} + 12x_{16} \leq 94.$$

$x_i \in \{0,1\}$ for all $i \in N$.

Now let $E_1 = \{1,2,3,4\}$, $E_2 = \{5,6,7,8,9,10,11\}$ and $E_3 = \{12,13,14,15,16\}$. Note that E_1 and E_2 are both covers, but E_3 is not a cover. The potential extreme candidate points are found using part of Kubik's method. These points in EP are given in Table 4. The feasible points 1-18 that form EP are bolded, and the nonbolded points are not feasible but are part of Kubik's algorithm.

Using three replications of Bolton's SSL algorithm, the extreme feasible points are found along the E_1, E_2 and E_3 axes. So EP_1 = (5,0),(4,1),(3,3),(2,4),(1,5), and (0,5); EP_2 =

p	E_1	E_2	E_3	p	E_1	E_2	E_3	p	E_1	E_2	E_3	p	E_1	E_2	E_3
–	3	1	0	–	2	3	0	–	1	4	0	–	0	6	0
1	**3**	**0**	**0**	**2**	**2**	**2**	**0**	**6**	**1**	**3**	**0**	**12**	**0**	**5**	**0**
–	3	0	1	–	2	2	1	**7**	**1**	**3**	**1**	–	0	5	1
				3	**2**	**1**	**1**	–	1	3	2	**13**	**0**	**4**	**1**
				–	2	1	2	**8**	**1**	**2**	**2**	–	0	4	2
				4	**2**	**0**	**2**	–	1	2	3	**14**	**0**	**3**	**2**
				5	**2**	**0**	**3**	**9**	**1**	**1**	**3**	**15**	**0**	**3**	**3**
				–	2	0	4	–	1	1	4	–	0	3	4
								10	**1**	**0**	**4**	**16**	**0**	**2**	**4**
								11	**1**	**0**	**5**	–	0	2	5
								–	1	0	6	**17**	**0**	**1**	**5**
												–	0	1	6
												18	**0**	**0**	**5**
												–	0	0	6

Table 4: Potential extreme candidates points on axes with feasible points bolded

(3,0),(2,3),(1,5), and (0,5); and EP_3 = (3,0),(2,2),(1,3), and (0,5). Bolton's SSL extreme points on the axes are listed below in the order they would be found around P^{ch^3}, assuming an arbitrary starting point of (0,5,0).

p^{12}=(0,5,0), $adj_r^{p^{12}} = p^{15}$, $adj_l^{p^{12}} = p^2$.

p^2=(2,2,0), $adj_r^{p^2} = p^{12}$, $adj_l^{p^2} = p^1$.

p^1=(3,0,0), $adj_r^{p^1} = p^2$, $adj_l^{p^1} = p^5$.

p^5=(2,0,3), $adj_r^{p^5} = p^1$, $adj_l^{p^5} = p^{11}$.

p^{11}=(1,0,5), $adj_r^{p^{11}} = p^5$, $adj_l^{p^{11}} = p^{18}$.

p^{18}=(0,0,5), $adj_r^{p^{18}} = p^{11}$, $adj_l^{p^{18}} = p^{17}$.

p^{17}=(0,1,5), $adj_r^{p^{17}} = p^{18}$, $adj_l^{p^{17}} = p^{15}$.

p^{15}=(0,3,3), $adj_r^{p^{15}} = p^{17}$, $adj_l^{p^{15}} = p^{12}$.

These points are ordered according to their position around the convex hull. Together, they form the extreme adjacencies of a Eulerian tour for this example. So, EP contains the points along the axes $p^{12} = (0, 5, 0)$, $p^2 = (2, 2, 0)$, $p^1 = (3, 0, 0)$, $p^5 = (2, 0, 3)$, $p^{11} = (1, 0, 5)$, $p^{18} = (0, 0, 5)$, $p^{17} = (0, 1, 5)$, $p^{15} = (0, 3, 3)$, along with the interior points $p^3 = (2, 1, 1)$, $p^7 = (1, 3, 1)$, $p^8 = (1, 2, 2)$ and $p^9 = (1, 1, 3)$. The order of the interior points is arbitrary at this point.

Now CATS' main step begins with $p^{12} = (0, 5, 0)$. So let $p^m := p^{12}$ and so $p^r := p^{15} = (0, 3, 3)$ and $p^l := p^2 = (2, 2, 0)$. Note these are the adjacent points to p^{12}. Using these three points, a candidate plane, T', is solved using the system of equations.

$$\alpha_{E_1} 0 + \alpha_{E_2} 5 + \alpha_{E_3} 0 = 1,$$

$$\alpha_{E_1} 2 + \alpha_{E_2} 2 + \alpha_{E_3} 0 = 1,$$

$$\alpha_{E_1} 0 + \alpha_{E_2} 3 + \alpha_{E_3} 3 = 1.$$

which gives the values

$$d = (0(2 * 3 - 3 * 0) + 2(3 * 0 - 5 * 3) + 0(5 * 0 - 2 * 0)) = -30,$$

$$\alpha_{E_1} = (5(0 - 3) + 2(3 - 0) + 2(0 - 0))/(-30) = \tfrac{3}{10},$$

$$\alpha_{E_2} = (0(0 - 0) + 0(2 - 0) + 3(0 - 2))/(-30) = \tfrac{1}{5},$$

$$\alpha_{E_3} = (0(2 - 3) + 2(3 - 5) + 0(5 - 2))/(-30) = \tfrac{2}{15}.$$

So $\alpha_{E_1} = \frac{3}{10}, \alpha_{E_2} = \frac{1}{5}$, and $\alpha_{E_3} = \frac{2}{15}$. In the form of an inequality, this cutting plane, T', $\frac{3}{10}y_1 + \frac{1}{5}y_2 + \frac{2}{15}y_3 \leq 1$ is tested against all points in EP. Evaluating $p^1 = (3, 0, 0)$ results in

$\frac{3}{10} * 3 + \frac{1}{5} * 0 + \frac{2}{15} * 0 = \frac{9}{10} \leq 1$. Continuing through all points in EP, it is determined that the point $p^7 = (1, 3, 1)$ violates this potential inequality $\frac{3}{10} * 1 + \frac{1}{5} * 3 + \frac{2}{15} * 1 = 31/30 > 1$. So, let $p^i := p^7$. The point p^r is dropped and p^i is added to form a new plane.

Because p^7 violated T', a new plane is created to accommodate the violated point. Let T' contain $p^{12} = (0, 5, 0), p^2 = (2, 2, 0), p^7 = (1, 3, 1)$, and the inequality is $\frac{3}{10}y_1 + \frac{1}{5}y_2 + \frac{1}{10}y_3 \leq 1$. This plane is tested for validity against all points in EP and is valid because all points in EP satisfy this inequality. Thus, T' becomes T^1, and the subroutine extreme plane is called to check for any more points meeting the plane at equality. None are found, and so the edge between p^2 and p^{12} is replaced with p^2 to p^7 and p^7 to p^{12}, which completes the Eulerian tour. Thus, the adjacency sets for p^{12} and p^2 are updated to include p^7. So $adj_l^{12} := p^7$, $adj_r^2 := p^7$, $adj_r^7 := p^{12}$, and $adj_l^7 := p^2$.

Now, set $T' = p^{12}, p^{15}, p^7$, with $p^m = p^{12}, p^l = p^7$ and $p^r = p^{15}$ and a plane equation of $\frac{4}{15}y_1 + \frac{1}{5}y_2 + \frac{2}{15}y_3 \leq 1$. This plane is checked against all extreme points, and since no points are violated, it is determined to be valid as well. Therefore $T^2 = T'$. The extreme plane routine checks for any points in EE meeting this plane at equality. In this instance, no others do, and so it is known that this is the final plane to use the point p^{12}. The adjacent points for each are updated as follows: $adj_l^{15} := p^7, adj_r^7 := p^{15}$, and p^{12} is removed from EP. Figure 10 depicts this scenario.

The algorithm now continues through the convex hull using the next extreme point, p^2. Note this is the original adjacent point left for p^{12}. Let $p^m := p^2$, with adjacent points $adj_r^{p^2} := p^7$ and $adj_l^{p^2} := p^1$. Now, $T' = p^2, p^7, p^1$, with plane $\frac{1}{3}y_1 + \frac{1}{6}y_2 + \frac{1}{6}y_3 = 1$. However, p^5 violates this inequality. Let $p^i := p^5$, and so now T' contains p^2, p^1, and p^5 with plane

$\frac{1}{3}y_1 + \frac{1}{6}y_2 + \frac{1}{9}y_1 = 1$. This plane violates no other points and so is called T^3. The extreme plane subroutine does not find a new point that meets it at equality, and so p^5 may be added to the adjacency sets for p^2 and p^1. The next T' from p^2 to be checked is comprised of the points p^2, p^5, and p^7, and its equation $\frac{5}{16}y_1 + \frac{3}{16}y_2 + \frac{1}{8}y_3 = 1$ violates no points and is valid. This plane $T^4 := T'$, and the extreme plane subroutine finds no new points meeting it at equality.

To demonstrate a case where EE is nontrivial, consider the plane found at the next p^m. T' is made up of the points $p^m := p^5, p^r := p^7$ and $p^l := p^{11}$ with plane $\frac{2}{7}y_1 + \frac{4}{21}y_2 + \frac{1}{7}y_3 = 1$. It violates no points in EP, so $T^5 := T'$. However, p^{15} meets T^5 at equality and is the only other point in EE besides p^5, p^7 and p^{11}. This is an example of a plane with more than three integer points that meet it at equality. For the purpose of this algorithm, p^{15} must be determined to be either an extreme point or not. CATS creates three temporary planes, as shown in Table 5. An r' value approximately equal to $\frac{1}{9} - \epsilon$ is used, where ϵ is a small irrational number that can be ignored in the calculations of this example. Let W' be the temporary plane created for each step in the extreme plane routine. The values for each of the points in Table 5 are found by inserting the components of each point into the equation for the temporary planes (i).

To find the adjacency from $p^{11} = adj_l^{p^m}$, let $q^m := p^{11}$ and $q^r := p^m = p^5$. The first temporary plane, W^1, is composed of the points $q^m,\ p^{15}$ and (r', r', r'). Evaluating p^5 and p^7 in this plane results in $\frac{191}{15}$ and $\frac{191}{15}$, respectively. Since both are strictly greater than 1, p^{15} is a candidate adjacent point to p^{11}. The second temporary plane, W^2, is composed of the points $q^m,\ p^7$ and (r', r', r'). Evaluating p^5 and p^{15} in this plane results in 23 and

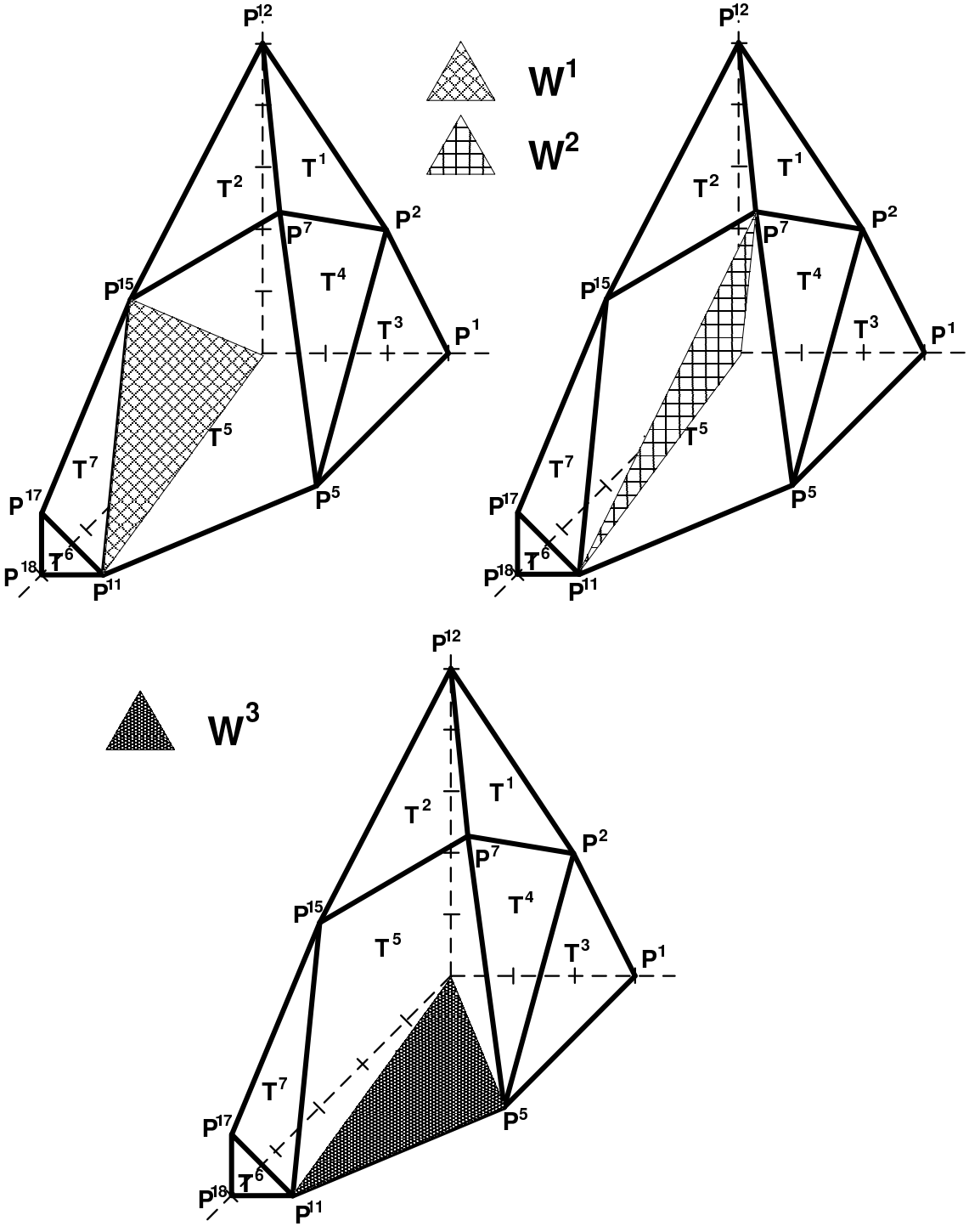

Figure 12: Temporary planes W' created in the extreme plane routine

Temp	Points	α'_{E_1}	α'_{E_2}	α'_{E_3}	p	Value	p	Value	Conclusion
W^1	$p^{11}, p^{15}, (r', r', r')$	$\frac{26}{3}$	$\frac{28}{15}$	$-\frac{23}{15}$	p^5	$\frac{191}{15}$	p^7	$\frac{191}{15}$	$p^j := p^{15}$
W^2	$p^{11}, p^7, (r', r', r')$	16	-4	-3	p^5	23	p^{15}	-21	$p^j :\neq p^7$
W^3	$p^5, p^{11}, (r', r', r')$	$\frac{2}{7}$	$\frac{60}{7}$	$\frac{1}{7}$	p^{15}	$\frac{183}{7}$	p^7	$\frac{183}{7}$	$p^j :\neq p^5$

Table 5: Temporary plane points, coefficients, values and outcomes

-21, respectively. Thus, p^7 is not a candidate extreme point adjacent to p^{11}. Finally, the temporary plane, W^3, is composed of the points q^m, p^5 and (r', r', r'). Evaluating p^7 and p^{15} in this plane results in $\frac{183}{7}$ and $\frac{183}{7}$, respectively. However, q^r, which is $p^r = p^m$, is on this plane and so p^5 is not a left candidate extreme point adjacent to p^{11}. Thus, q^m's left extreme point is p^{15}. A graphic of this routine is illustrated in Figure 12.

Now $q^m := p^{15}$ and $q^r := p^{11}$. This process continues to identify the adjacent extreme points on T^5. The next extreme points, p^7 and then p^5, complete the Eulerian tour as the adjacent points are updated. Thus, the original edge between p^{11} and p^5 is replaced with the edges from p^{11} to p^{15}, p^{15} to p^7, and p^7 to p^5. Notice that p^{15} previously contained adjacent points, and these must be retained during the updating process.

CATS continues by selecting the next extreme point, $p^m := p^{11}$. T' is p^{11}, p^{18} and p^{15}, which is violated by p^{17}. T' is now p^{11}, p^{18} and p^{17}, yielding a valid inequality $T^6 = 0y_1 + 0y_2 + \frac{1}{5}y_3 \leq 1$. T^7 is confirmed through p^{11}, p^{17} and p^{15}, resulting in $\frac{1}{6}y_1 + \frac{1}{6}y_2 + \frac{1}{6}y_3 \leq 1$ as the valid inequality.

T^7 is the last valid inequality to define P^{ch^3}. From here on, CATS continues finding duplicate inequalities using the remaining extreme points until the algorithm terminates.

Figure 13 illustrates the list of valid inequalities as they define the convex hull. The valid inequalities produced from CATS for P^{ch^3} with their respective points are listed below. Additionally, the valid inequalities for P^{ch}_{KP} in the form of $\alpha_{E_1} \sum_{i \in E_1} x_i + \alpha_{E_2} \sum_{i \in E_2} x_i + \alpha_{E_3} \sum_{i \in E_3} x_i \leq 1$ are listed immediately following.

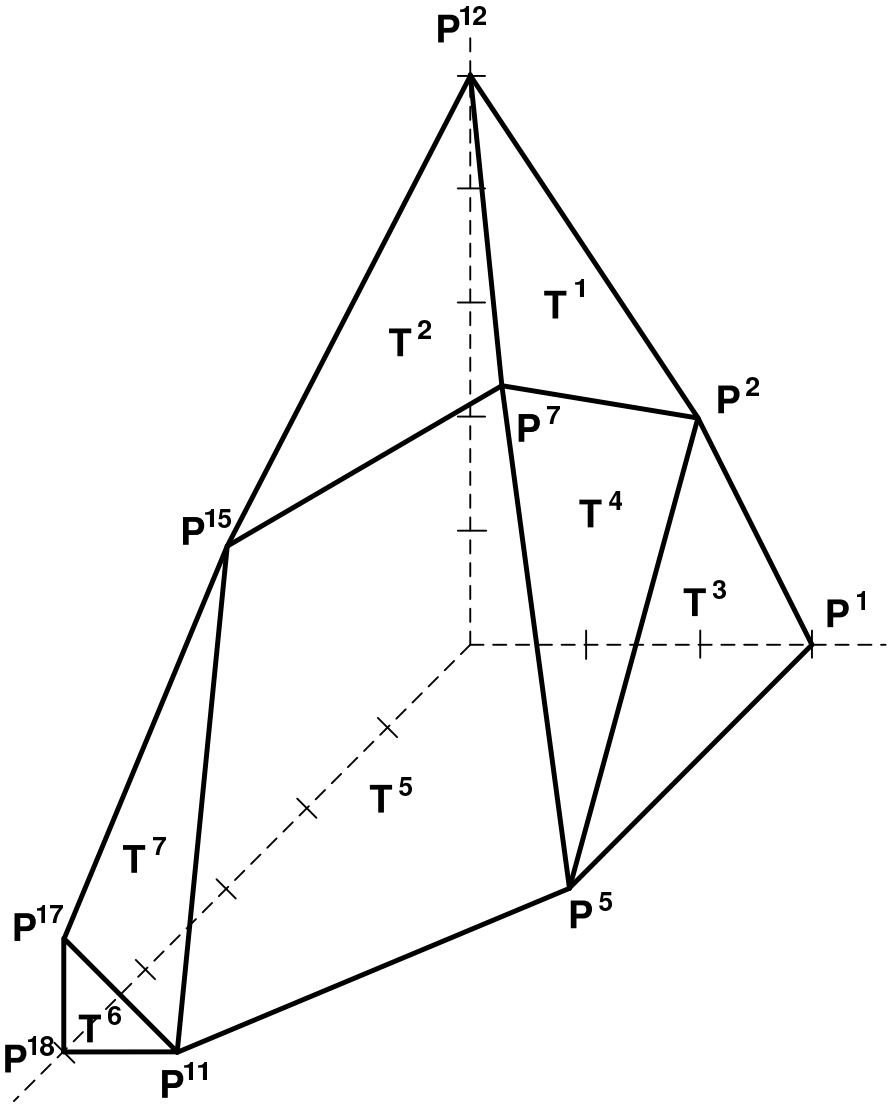

Figure 13: Valid inequalities T' that define P^{ch^3}

$T^1 : p^{12} = (0, 5, 0), p^2 = (2, 2, 0), p^7 = (1, 3, 1)$ is $\frac{3}{10}y_1 + \frac{1}{5}y_2 + \frac{1}{10}y_3 \leq 1$

$\frac{3}{10} \sum_{i \in E_1} x_i + \frac{1}{5} \sum_{i \in E_2} x_i + \frac{1}{10} \sum_{i \in E_3} x_i \leq 1.$

$T^2 : p^{12} = (0,5,0), p^7 = (1,3,1), p^{15} = (0,3,3)$ is $\frac{4}{15}y_1 + \frac{1}{5}y_2 + \frac{2}{15}y_3 \leq 1$

$$\tfrac{4}{15}\textstyle\sum_{i\in E_1} x_i + \tfrac{1}{5}\sum_{i\in E_2} x_i + \tfrac{2}{15}\sum_{i\in E_3} x_i \leq 1.$$

$T^3 : p^2 = (2,2,0), p^1 = (3,0,0), p^5 = (2,0,3)$ is $\frac{1}{3}y_1 + \frac{1}{6}y_2 + \frac{1}{9}y_3 \leq 1$

$$\tfrac{1}{3}\textstyle\sum_{i\in E_1} x_i + \tfrac{1}{6}\sum_{i\in E_2} x_i + \tfrac{1}{9}\sum_{i\in E_3} x_i \leq 1.$$

$T^4 : p^2 = (2,2,0), p^5 = (2,0,3), p^7 = (1,3,1)$ is $\frac{5}{16}y_1 + \frac{3}{16}y_2 + \frac{1}{8}y_3 \leq 1$

$$\tfrac{5}{16}\textstyle\sum_{i\in E_1} x_i + \tfrac{3}{16}\sum_{i\in E_2} x_i + \tfrac{1}{8}\sum_{i\in E_3} x_i \leq 1.$$

$T^5 : p^5 = (2,0,3), p^{11} = (1,0,5), p^7 = (1,3,1)$ is $\frac{2}{7}y_1 + \frac{4}{21}y_2 + \frac{1}{7}y_3 \leq 1$

$$\tfrac{2}{7}\textstyle\sum_{i\in E_1} x_i + \tfrac{4}{21}\sum_{i\in E_2} x_i + \tfrac{1}{7}\sum_{i\in E_3} x_i \leq 1.$$

$T^6 : p^{11} = (1,0,5), p^{18} = (0,0,5), p^{17} = (0,1,5)$ is $0y_1 + 0y_2 + \frac{1}{5}y_3 \leq 1$

$$0\textstyle\sum_{i\in E_1} x_i + 0\sum_{i\in E_2} x_i + \tfrac{1}{5}\sum_{i\in E_3} x_i \leq 1.$$

$T^7 : p^{11} = (1,0,5), p^{17} = (0,1,5), p^{15} = (0,3,3)$ is $\frac{1}{6}y_1 + \frac{1}{6}y_2 + \frac{1}{6}y_3 \leq 1$

$$\tfrac{1}{6}\textstyle\sum_{i\in E_1} x_i + \tfrac{1}{6}\sum_{i\in E_2} x_i + \tfrac{1}{6}\sum_{i\in E_3} x_i \leq 1.$$

These inequalities are all valid in P^{ch}_{KP} due to Lemma 3.1. To determine if any or all of these seven inequalities are facet defining, check with Theorem 3.4. The inequality from T^4 is selected for demonstration because of the interesting alpha values, but this procedure is repeated for the other inequalities.

The three points from T^4 are p^2, p^5 and p^7. From the theorem, let $p^1 := p^7$ which is $(1,3,1)$, so $p^1_1 = 1, p^1_2 = 3$, and $p^1_3 = 1$. Let $p^2 := p^2$ which is $(2,2,0)$, so $p^2_1 = 2, p^2_2 = 2$,

and $p_3^2 = 0$. Finally, $p^3 := p^5$ which is $(2, 0, 3)$, so $p_1^3 = 2, p_2^3 = 0$, and $p_3^3 = 3$. Figure 14 illustrates the affinely independent points for this inequality. In the figures of affinely independent points, lines have been drawn to help the reader visualize the conditions of the theorems and how the elements are arranged for each of the three points.

In the first column, notice how the first point, contains one element from E_1, three elements from E_2, and one element from E_3. The element is the largest in E_1, and so once this point is valid, the remaining three points are valid as well. This follows condition i of Theorem 3.4, where $p_1^1 = 1$. The elements in E_2, $i^2_{e_2-p_2^{t^1}+1}, ..., i^2_{e_2}$, and E_3, $i^3_{e_3-p_3^{t^1}+1}, ..., i^3_{e_3}$, correspond to the lightest elements, p_2^i and p_3^i, in E_2 and E_3 that the hiker can carry. Thus point $(1, 3, 1)$ has four affinely independent points.

Through a similar procedure, $(2, 2, 0)$ and $(2, 0, 3)$ are also confirmed to be affinely independent. Notice for both points $(2, 2, 0)$ and $(2, 0, 3)$, condition ii of Theorem 3.4 is employed, and only the first and last affinely independent points in each set require validation. These are the points that, once checked, confirm the feasibility of the other points.

This facet defining inequality is a new class of inequalities that CATS can efficiently find. First, rewrite $\frac{5}{16}\sum_{i\in E_1} x_i + \frac{3}{16}\sum_{i\in E_2} x_i + \frac{1}{8}\sum_{i\in E_3} x_i \leq 1$ to $5\sum_{i\in E_1} x_i + 3\sum_{i\in E_2} x_i + 2\sum_{i\in E_3} x_i \leq 16$. Since there are only 6 elements in E_3, this inequality is not based upon a cover inequality. Clearly, it is not possible to obtain this inequality using Zemel's or Bolton's method. Obviously, this inequality could be generated using simultaneous lifting, but you would have to have started with $5\sum_{i\in E_1} x_i + 3\sum_{i\in E_2} x_i \leq 16$ or one of the other two combinations and lift in the third set. No one except an oracle would know to choose such a bizarre inequality as a starting inequality and thus it can be concluded CATS can

find new classes of facet defining inequalities.

1	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	0	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	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	1	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	0	0	0	0	0	0	0	0
1	1	1	1	0	0	0	0	0	1	1	0	0	0	0	0
1	1	1	1	0	0	0	0	1	0	1	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	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	1	1	1
0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1
0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	1
1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	0

Figure 14: Affinely independent points for $\frac{5}{16}\sum_{i\in E_1} x_i + \frac{3}{16}\sum_{i\in E_2} x_i + \frac{1}{8}\sum_{i\in E_3} x_i \leq 1$

Theorem 3.4 is used to confirm four of the seven inequalities as facet defining, and as illustrated in Figure 13, these facet defining inequalities occur primarily across the center of P^{ch^3}. Below is a list of the facet defining inequalities and their affinely independent points.

$T^2 : p^{12} = (0,5,0), p^7 = (1,3,1), p^{15} = (0,3,3)$ is $\frac{4}{15}y_1 + \frac{1}{5}y_2 + \frac{2}{15}y_3 \leq 1$ Figure 15

$T^3 : p^2 = (2,2,0), p^1 = (3,0,0), p^5 = (2,0,3)$ is $\frac{1}{3}y_1 + \frac{1}{6}y_2 + \frac{1}{9}y_3 \leq 1$ Figure 16

$T^4 : p^2 = (2,2,0), p^5 = (2,0,3), p^7 = (1,3,1)$ is $\frac{5}{16}y_1 + \frac{3}{16}y_2 + \frac{1}{8}y_3 \leq 1$ Figure 14

$T^5 : p^5 = (2,0,3), p^{11} = (1,0,5), p^7 = (1,3,1)$ is $\frac{2}{7}y_1 + \frac{4}{21}y_2 + \frac{1}{7}y_3 \leq 1$ Figure 17

CATS' ability to systematically find new inequalities throughout P^{ch^3} makes this beneficial for removing part of the linear relaxation for P^{ch}_{KP}. Furthermore, this new class of

1	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	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	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	0	1	1	1	1	0	0	0	0	0
0	0	0	0	1	1	1	0	1	1	1	0	0	0	0	0
1	1	1	1	1	1	1	1	0	1	1	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	1	1	0	1	1	1	1	1
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	1	1	1
0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1
0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	1
1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	0

Figure 15: Affinely independent points for $\frac{4}{15}\sum_{i\in E_1} x_i + \frac{1}{5}\sum_{i\in E_2} x_i + \frac{2}{15}\sum_{i\in E_3} x_i \leq 1$

0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	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	0	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	1	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	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	1	0	1	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	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	1	1	1
0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1
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	1	1	1	1	0

Figure 16: Affinely independent points for $\frac{1}{3}\sum_{i\in E_1} x_i + \frac{1}{6}\sum_{i\in E_2} x_i + \frac{1}{9}\sum_{i\in E_3} x_i \leq 1$

1	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	0	0	0	0	0	0	0	0	1	1	1	1	1
0	0	0	1	0	0	0	0	0	0	0	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	1	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	1	1	1	0	0	0	0	0
1	1	1	1	0	0	0	1	0	1	1	0	0	0	0	0
1	1	1	1	1	1	1	1	1	0	1	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	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	1	1	1
0	0	0	0	1	1	1	1	1	1	1	0	1	0	1	1
0	0	0	0	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	0

Figure 17: Affinely independent points for $\frac{2}{7}\sum_{i\in E_1} x_i + \frac{4}{21}\sum_{i\in E_2} x_i + \frac{1}{7}\sum_{i\in E_3} x_i \leq 1$

inequalities would not have been found without CATS. For example, the inequality from T^4, $\frac{5}{16}y_1 + \frac{3}{16}y_2 + \frac{1}{8}y_3 \leq 1$, defines much of the center of P^{ch^3}. The likelihood of finding and choosing this inequality through traditional lifting methods is minimal, if not impossible. Therefore, CATS theoretically appears to be a useful computational tool. The proceeding chapter discusses the beneficial computational results of CATS.

4 CATS Computational Results

The contribution of this research is the application of synchronized simultaneous lifting in three sets and the development of an algorithm to systematically find a new class of valid inequalities. The purpose of this section is to describe the quantitative advancements of CATS to support its implementation into commercial integer programming software. The computational results show that CATS cuts are found relatively easily compared with the IP solving time, and CATS does decrease the solution time for some IPs.

The results of this computational study were obtained through the use of an Intel (R) Core i7 computer with a 1.58 GHz processor and 3.0 GB of RAM. A commercial optimization software, CPLEX [38], was used to compare the results with and without the CATS cuts. All results are reported in seconds. The code itself is created in C++.

The search for an appropriate class of problems attempted a variety of factors. The number of variables was considered, as well as the number of constraints, the range of constraint and objective function coefficients, size of the sets, analysis of gaps between sets, and even the reduced costs for each variable. An appropriate class of problems also has to solve within a general time interval. Problems solving too quickly are unlikely to show either positive nor negative results for CATS. Problems requiring numerous hours to solve or that exhausted the computers memory are also impractical for a computational study.

In the opinion of the author, no ideal class of problems was found to demonstrate the power of CATS inequalities. Further computational studies are encouraged to understand the precise computational benefits of this thesis.

The class of problems selected is a randomly generated multiple knapsack problem (MKP) with two constraints. An MKP is simply a KP with one or more additional constraints, and this may be formulated as Maximize $\sum_{i \in N} c_i x_i$ subject to $\sum_{i \in N} a_{ji} x_i \leq b_j$ for $j = 1, ..., r$ and $x_i \in \{0, 1\} \; \forall \; i \in N$ where r is the number of rows and $a_{ji} \geq 0$ for all $i \in N$ and $j \in \{1, ..., r\}$.

The constraint coefficients, a_{ij}, are random integers from a uniform distribution between 50,000 and 250,000. Each objective coefficient, c_i, is calculated by summing the column coefficients and adding on a random integer between 0 and u. Alternatively, $c_i = \sum_{j=1}^{r} a_{ji} + u'$, where u' is an integer taken from a $U(0, u)$ for all $j \in N$. The right hand side of each constraint is the sum of all a_{ji} in that constraint multiplied by s, where s is a slackness ratio and rounded down. Formally, $b_j = \lfloor s \sum_{i=1}^{n} a_{ji} \rfloor$ for $j = 1$ and 2.

The number of variables in each problem is dependent on the solution time using CPLEX. The problems that were neither trivial nor too demanding happened to have 50 and 55 variables. Additionally, in an effort to place variables in the sets that could have more of an impact on removing part of the linear relaxation, the reduced costs for all variables are sorted before the sets were chosen. The variables with reduced costs in the lowest 20% are removed from the list of variables to be placed into the sets.

The size of the sets E_1, E_2 and E_3 are made variable to remove nonuniform gaps from within a particular set, with each set confined to a range of values appropriate for the size of the problem. The coefficients are sorted in descending order, and so generally the magnitude of E_1 is smaller than E_2, and E_2 is smaller still than E_3. With a minimum and maximum set size predetermined, a subroutine seeks the largest gap between constraint coefficients within that data range. The minimum size of E_1 is 15% of the total number of variables,

with a maximum value of 30%. The set E_2 contains between 15% and 30%, and E_3 contains between 15% and the maximum number of variables remaining.

The preprocessing time for CATS is calculated by running 300 problems with one constraint without solving the integer program. The cumulative time is approximately 9.6 seconds, which gives an average time of .032 seconds per problem.

The first of the computational tests listed in Table 6 are for problems with $s = .25$ and $u = 4$. This set of 20 randomly generated problems contains 50 variables and uses two knapsack constraints. The preprocessing time is included in the recorded times, but it is essentially negligible with CATS average runtime at less than one thirtieth of a second.

The second of the computational tests listed in Table 7 are 20 randomly generated problems, each containing 55 variables and using two knapsack constraints. With the values of $s = .125$ and $u = 5$, CATS preprocessing time for this set of problems nearly matches the previous set, even though the number of variables increased by 10%.

Both of these data sets illustrate the effectiveness of the previously undiscovered class of inequalities. CATS finds approximately 12 distinct inequalities per problem, with an average of 13.2 and 11.4 cuts from each set, respectively.

CATS finds these valid inequalities in a very short preprocessing time in this study. Although the computational study is concerned only with the problems tested, from observations and theorical results the preprocessing time for CATS is proportionally linked to the number of variables in the problem. For example, CATS requires approximately 6.3 seconds to generate cuts for 60 problems with 100 variables (not solve the IP using CPLEX). Even in these larger problems, the preprocessing time is reasonable with an average of .105 seconds.

Problem	$CPLEX$	$CATS$	$(CPLEX - CATS)$	$CATS\%$ *Improvement*	$CATS$ *Cuts*	*Winner*
1	508	427	81	15.9%	10	CATS
2	524	774	-250	-47.7%	12	CPLEX
3	349	449	-100	-28.7%	13	CPLEX
4	664	292	372	56.0%	13	CATS
5	561	457	104	18.5%	12	CATS
6	312	752	-440	-141.0%	13	CPLEX
7	433	414	19	4.4%	15	CATS
8	106	83	23	21.7%	13	CATS
9	1002	451	551	55.0%	19	CATS
10	187	148	39	20.9%	8	CATS
11	82	126	-44	-53.7%	15	CPLEX
12	420	177	243	57.9%	13	CATS
13	194	242	-48	-24.7%	10	CPLEX
14	641	778	-137	-21.4%	16	CPLEX
15	675	363	312	46.2%	13	CATS
16	133	368	-235	-176.7%	11	CPLEX
17	735	514	221	30.1%	15	CATS
18	538	190	348	64.7%	20	CATS
19	173	164	9	5.2%	13	CATS
20	238	707	-469	-197.1%	10	CPLEX
Average	**424**	**394**	**30**	**7.1%**	**13.2**	**12/20**

Table 6: Data reported for problems with 50 variables

Problem	$CPLEX$	$CATS$	$(CPLEX - CATS)$	$CATS\%$ *Improvement*	$CATS$ *Cuts*	$Winner$
1	1989	1338	651	32.7%	14	CATS
2	1309	1296	13	1.0%	8	CATS
3	1443	1179	264	18.3%	10	CATS
4	1097	927	170	15.5%	7	CATS
5	1494	834	660	44.2%	12	CATS
6	2208	2138	70	3.2%	10	CATS
7	1137	1152	-15	-1.3%	7	CPLEX
8	1174	955	219	18.7%	17	CATS
9	1020	1209	-189	-18.5%	11	CPLEX
10	1033	1078	-45	-4.4%	8	CPLEX
11	1450	1191	259	17.9%	13	CATS
12	1465	1022	443	30.2%	9	CATS
13	1268	1638	-370	-29.2%	18	CPLEX
14	1423	1636	-213	-15.0%	9	CPLEX
15	703	796	-93	-13.2%	14	CPLEX
16	1907	1621	286	15.0%	12	CATS
17	1089	1286	-197	-18.1%	13	CPLEX
18	1179	1019	160	13.6%	14	CATS
19	1501	841	660	44.0%	7	CATS
20	1718	1531	187	10.9%	14	CATS
Average	**1380**	**1234**	**146**	**10.6%**	**11.4**	**13/20**

Table 7: Data reported for problems with 55 Variables

CATS is capable of reducing the processing time for some integer programs by an average of 8.9% or more. The runtime was reduced by 7.1% in the first data set and by 10.6% in the second data set. In this second set of problems, CATS cut nearly 50 minutes off of the 7.5 hour runtime. Given the greater length of the runtime for these problems and many integer programs in general, CATS is able to maintain a runtime advantage for a wide range of problems.

Further computational studies were performed with simpler problems requiring a total between one and two minutes to solve. These results are even more impressive, as CATS reduced these processing times by at least 30% and in some cases over 60%. Table 8 presents data for 10 problems with a CATS runtime 51.5% better than CPLEX.

Problem	*CPLEX*	*CATS*	$(CPLEX - CATS)$	*CATS% Improvement*	*CATS Cuts*	*Winner*
1	1	2	-1	-100.0%	16	CPLEX
2	1	1	0	0.0%	14	-
3	6	2	4	66.7%	10	CATS
4	22	1	21	95.5%	9	CATS
5	2	6	-4	-200.0%	7	CPLEX
6	5	3	2	40.0%	12	CATS
7	17	11	6	35.3%	9	CATS
8	12	11	1	8.3%	11	CATS
9	21	9	12	57.1%	12	CATS
10	10	1	9	90.0%	9	CATS
Average	**9.7**	**4.7**	**5**	**51.5%**	**10.7**	**7/10**

Table 8: Short runtime problems with 50 variables

CATS valid inequalities with the most pronounced impact on the reduction in processing time are the inequalities spanning across the center of the P^{ch^3}. For example, one of this problems produced the inequality $11\sum_{i\in E_1} x_i + 7\sum_{i\in E_2} x_i + 4\sum_{i\in E_3} x_i \leq 60$, which spans

much of P^{ch^3}. The points in EE that meet this inequality at equality are: (1,7,0), (3,3,1), (3,2,3), (2,2,6) and (0,4,8). This is a large number of points in EE, indicating the effectiveness of this cutting plane. Notice that this inequality could not be found with previous lifting methods.

Overall, this computational study shows CATS is indeed fast in finding effective inequalities, with the average preprocessing time of less than .032 seconds. The resulting CATS inequalities can reduce the solution time of integer programming problems by about 8.9% over CPLEX.

5 Conclusion

The purpose of this research was to augment synchronized simultaneous lifting from two to three dimensions. The Cutting-plane Algorithm in Three Sets does obtain valid inequalities through systematically updating adjacent points and creating new inequalities when a violated point is found. Due to the methodical process for identifying valid inequalities from an extreme point and maintaining the Eulerian tour, CATS is able to find all nontrivial facet defining inequalities in P^{ch^3}. These inequalities are then converted to valid inequalities for P^{ch}_{KP}.

CATS finds new classes of cutting planes for P^{ch}_{KP} that cannot be found using any previous methods without the help of an omniscient being. Furthermore, theoretical results show that CATS runs in polynomial effort and that there are easily checkable conditions for CATS' inequalities to be facet defining in P^{ch}_{KP}.

The computational results support a significant reduction in solution time compared with a commercial optimization software. The problems with CATS cuts solved 8.9% faster than with standard CPLEX. The processing time for CATS is minimal at approximately .032 seconds. Given its theoretical worst case time of quartic effort times the number of inequalities generated, CATS is a viable technique to be implemented to solve large integer programs.

5.1 Future Work

Both CATS and synchronized simultaneous lifting are in their infancy and thus there remains substantial research to be done in this area. This section provides the reader with a few of these ideas.

There needs to be a greater computational understanding into the precise uses of three set synchronized simultaneous lifting. This research primarily focused on the theoretical advancements of CATS, and no ideal class of problems was found to demonstrate CATS true power. It is the opinion of the author even greater computational results can be obtained through selecting a different problem class and providing better guidelines for the input sets. Additionally, CATS should be compared to benchmark knapsack problems to determine its effectiveness against commonly studied problems.

An obvious future research idea is to expand synchronized simultaneous lifting to more than three sets. This would require adjacency lists in a minimum of four dimensions, and therefore graphical illustrations would no longer be feasible. The complexity of creating CANS (Cutting-plane Algorithm in n Sets) would be significantly more than two and even three sets. However, with the significant computational benefits of growing from two to three sets would indicate that CATS in higher numbers of sets would provide even more reduction in runtime.

CATS should be applied to problems where a cover inequality is given as the starting inequality. Instead of simply finding valid inequalities from nothing, a cover inequality could allow CATS to seek different inequalities on P^{ch}. Thus, CATS would become a type of synchronized simultaneous up lifting.

CATS should also be applied to other classes of integer programs, including non-binary problems, general mixed integer problems, and multiple constraint knapsack problems. Such theoretical advancements could vastly improve CATS and make the computational results much stronger.

Pursuing these research topics should enable CATS to have a lasting impact on integer programming research. Ultimately, the implementation of CATS into commercial code would solidify the lasting impact of this research.

6 Afterthoughts

In some sense, the development of CATS has followed the progression of Kansas State University's Men's Basketball Program during the my college tenure (2006-2010). This research began with hand drawn diagrams and an unknowingness of the author as to the complexities of synchronized simultaneous lifting in three dimensional space. Similarly, KSU's basketball program did not qualify for the NCAA tournament in 2007. However, every challenge was met with determination and new discoveries for both CATS, which slowly turned an idea into a thesis and athletic potential into an Elite Eight appearance.

In my opinion, both CATS and the K-State's Men's Basketball team have the potential to be remembered forever. The requirements are simply stated, but difficult to achieve. The Wildcats need a national championship and CATS needs to be implemented into commercial code. Other than dedicated support, I have no worthwhile recommendations as to how Coach Martin should improve his team to win a championship. However, the future research section describes several avenues with the potential to help take CATS to the next level.

On a personal note, each time I hear Dave Lewis yell "BRING ON THE CATS!" and the fans cheer "GO CATS", a smile will cross my face. Unknowingly, each fan not only encourages Kansas State athletics, but is also cheering on my contribution to the field of integer programming, Generating an Original Cutting-plane Algorithm in Three Sets (GO CATS).

References

[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**, (2), May 2003, 170-82.

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

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

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

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

[7] 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.

[8] 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.

[9] Bolton, Jennifer (2009). "Synchronized simultaneous lifting in binary knapsack polyhedra," *MS Thesis*, Department of Industrial and Manufacturing Systems Engineering, Kansas State University.

[10] 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.

[11] Constantino, et al. (2008). "A new mixed-integer programming model for harvest scheduling subject to maximum area restrictions, *Operations Research* **56**, 542-551.

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

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

[14] 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, 100-9.

[15] 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.

[16] Euler, L., "Solutio problematis ad geometriam situs pertinentis", *Comment. Academiae Sci. I. Petropolitanae*, **8** (1736), 128-140.

[17] 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.

[18] Finn, Frank J. (1973). "Integer programming, linear programming and capital budgeting." *Abacus*, **13**, 180-192.

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

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

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

[22] 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.

[23] Iwamura, K. and B. Liu (1999). "Dependent-chance integer programming applied to capital budgeting." *Journal of the Operations Research Society of Japan*, **11**, 117-127.

[24] 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.

[25] 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, 431-40.

[26] Kolliopoulos S. and P. Ilissia.(2007), "Partially ordered knapsack and applications to scheduling ," *Discrete Applied Mathematics archive* **155** (8), 889-897.

[27] Kubik, Lauren, (2009) "Simultaneously lifting multiple sets in binary knapsack integer programs," *MS Thesis*, Department of Industrial and Manufacturing Systems Engineering, Kansas State University.

[28] 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.

[29] 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.

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

[31] 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.

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

[33] 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.

[34] 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.

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

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

[37] Subramanian, R., Scheff R., Quillinan J., Wiper D. S., and Marsten R. (1994) "Cold-start: fleet assignment at Delta Air Lines." *INTERFACES*, **24**, January-February 1994, 104-120.

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

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

[40] Trick M. A. (2004), "Using sports scheduling to teach integer programming," *INFORMS Transactions on Education*, **5** (1).

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

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

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

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

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