Abstract

This thesis explores a geometric structure called the linear ordering polytope. The linear ordering polytope, $L^n$, is the convex hull of a collection of vertices constructed from every permutation of a set of size $n$. Because the number of vertices of $L^n$ grows quickly with the size of the ordering and because the combinatorics underlying their configuration is complex, the complete set of facets of the general linear ordering polytope is not known.

We explore the facets of $L^n$ by looking at a connection between the solutions of a class of linear programs based in voting theory and the facets of $L^n$. We begin with expository work in polyhedral theory, optimization, and voting theory, and finish with an explanation of the connection between the linear programs and $L^n$.

Since the linear ordering polytope appears in a wide range of problems that arise in discrete optimization, this work has possible practical implications for the design of efficient algorithms.
Voting Tournaments and
The Linear Ordering Polytope

Grace E. Barkhuff
Under Direction of Dylan Shepardson

May 2016

A thesis submitted to the department of mathematics and statistics in partial fulfilment of the requirements for the degree of Bachelor of Arts in Mathematics at Mount Holyoke College.
Acknowledgements

I want to thank Dylan Shepardson, my thesis advisor, academic advisor, and mentor for the support he has given me my entire time at Mount Holyoke. I would never have been a math major if it weren’t for his Calc II class and all the talk of honeybees. Thank you for believing in me enough to “drag me down the rabbit hole” of the linear ordering polytope.

I want to thank Larry Schipull, for inspiring me to be my best, but to also be myself.

I want to thank my thesis committee for their support with this project.

I want to thank everyone at Carleton SMP ’14 for giving me the confidence to call myself a “mathematician,” and for everyone at BSM Fall ’14 for showing me how much passion, drive, and rigour that can mean. Thanks to both groups for welcoming me into their fantastic, wacky, math community.

I want to thank my family for loving and supporting me on any path I choose, and my friends for being willing to engage in dinner-time discussions on “Is math created or discovered?”

I want to thank the entire Mount Holyoke mathematics faculty, staff, and students for making the “Fourth Floor of Clapp” feel like home for the past four years.
Contents

1 Introduction 3

1.1 Motivation: Computational Complexity 7

1.2 Motivation: Classes of Facets 9

1.3 Remarks 10

2 Polyhedral Theory 11

2.1 Extreme Points 20

2.2 Polyhedra 23

2.3 Faces and Facets 28

2.4 The Wheel 30

3 The Linear Ordering Polytope 45
CONTENTS

4 LP, IP, and \( L^n \) 51

4.1 Acyclic Tournaments and \( L^n \) 51

4.2 The Linear Ordering Problem 53

4.3 TSP 54

4.4 Linear Programming Basics 59

4.5 Integer Programming from Linear Programming 62

5 LPs and \( L^n \) 70

5.1 The Correspondence when \( n = 3 \) 75

5.2 Facets of \( L^n \) 80

6 Conclusion 86

6.1 Future Work 86

6.2 Summary 88
Chapter 1

Introduction

“It is far beyond the scope of such investigations [as Grötschel et. al completed in 1985] to give a complete and general description of the linear-ordering polytope.” (Reinelt, 1993) [13]

In this thesis, we explore the geometries of a particular class of polytopes called linear ordering polytopes, $L^n$. These polyhedra are defined by the set of all possible ways to permute (or order) a set of elements.

Given a set $S$ of cardinality $n$, a permutation of $S$ is a function mapping $S$ to $S$ which is both one-to-one and onto. There are $n!$ such permutations, corresponding to the $n!$ ways to order the elements of $S$.

A permutation, $\pi : S \rightarrow S$, can be represented by a vector, $x$, in $\mathbb{R}^{2(n)}$ as
follows:

\[
x_{ij} = \begin{cases} 
1, & \text{if } \pi(i) > \pi(j). \\
0, & \text{otherwise.}
\end{cases}
\]

The \( n! \) permutations specify \( n! \) distinct vectors in \( \mathbb{R}^{2(n)} \). We call the convex hull of these vectors the \textit{linear ordering polytope}, \( L^n \).

A polyhedron in \( \mathbb{R}^m \) is a set of elements in \( \mathbb{R}^m \) defined by a finite number of inequality constraints. The linear ordering polytope is a particular polyhedron: it is bounded, and thus a polytope, and the convex hull of the permutation vectors implicitly defines the finite number of inequality constraints that bound it.[11]

What we want to know about \( L^n \) is the facet geometry of the polytope. In simple terms, the facets are the “sides” of the polyhedron. They are inequalities representing the boundary of the polyhedron.[1]

In this paper, we demonstrate a direct relationship between the facets of \( L^n \) and a specific voting optimization problem. This problem uses a type of voting system called a ranked voting system. In a ranked voting system, individuals submit a list of the candidates ranked in order of preference. This can be done by submitting a list, or permutation, of every candidate, where the leftmost candidate in the list is most preferred by the voter and the rightmost is least preferred. We call this list a ballot.

Any ballot can be uniquely represented by a voting tournament. A voting tournament is a complete directed graph where each vertex represents a
candidate in the election, and a single edge is directed *toward* the preferred candidate for every pair of candidates. One example of a voting tournament with three candidates—A, B, C— is in Figure 1.1.

This voting tournament represents the ballot (B, A, C). We see there is an edge directed from C to A and from C to B, indicating that C is the least preferred candidate. Additionally, there is an edge directed from both A and C toward B, indicating that B is the most preferred candidate. The tournament can be created with a single ballot, or unanimous agreement among voters.

On the other hand, not every voting tournament can be represented by a single ballot. Consider the tournament in Figure 1.2.

In a pairwise election, this tournament shows that A is preferred to C, B is preferred to A, and C is preferred to B. This creates a cycle. This type of voting tournament is called a “Condorcet paradox”, or “voting paradox” since it is impossible for a single ballot, or permutation, to create a
However, if we take the aggregate of three ballots/permutations, we can create this cycle. Those permutations are $(C, B, A)$, $(B, A, C)$ and $(A, C, B)$. Looking at all three permutations, we see $B$ is preferred to $A$ $\frac{2}{3}$ of the time, $C$ is preferred to $B$ $\frac{2}{3}$ of the time, and $A$ is preferred to $C$ $\frac{2}{3}$ of the time, creating this cycle.

This tournament cannot be constructed from a single permutation and unanimous agreement, but it can be constructed from three ballots and $\frac{2}{3}$ agreement.

When looking for agreement in a tournament, we must consider every pairwise election that is represented by the tournament. So, even if we are given a “mostly” acyclic tournament, but it contains the $3$–cycle voting paragraph as a subgraph, we would have to say the entire graph cannot be created with unanimous agreement, but with $\frac{2}{3}$ agreement.

Given a voting tournament, the optimization problem searched for the
maximum possible agreement that can result in that tournament.\cite{15}

1.1 Motivation: Computational Complexity

An application of finding the facets of $L^n$ is the Linear Ordering Problem (LOP). The problem is, given an algorithm to calculate a score for each permutation of a set $S$, what is the permutation giving the maximum score?

The LOP is known to be $NP$-complete, meaning it is unlikely that there is an algorithm that efficiently solves the linear ordering problem.\cite{17}

The brute force method of solving a particular LOP would be to enumerate all $n!$ permutations over the set $S$, where $|S| = n$, calculate the score for each, then choose the largest. As $n$ increases, the computational run-time of this method increases exponentially, making it inefficient to solve. Thus, alternative solution methods need to be considered.

One avenue of better methods are linear programming methods. The Simplex Method for linear programming, although formally not efficient in a worst-case analysis, is used in practice to find quick and exact solutions to optimization problems.\cite{1} The issue with using the Simplex Method for the LOP is that the LOP is an integer programming problem, requiring every variable to be binary. The Simplex Method will find the maximum possible

\footnote{In computational complexity theory, a problem belongs to the class $NP$ if a solution can be verified in polynomial time (i.e. if there exists a polynomially verifiable “certificate”). A problem is said to be $NP$-hard if every problem in the class $NP$ can be reduced to it in polynomial time. A problem is said to be $NP$-complete if it both belongs to the class $NP$ and is $NP$-hard. See \cite{16}, Chapter 7.}
score, but it is not restricted to giving integer solutions, and it may give a solution with fractional components.

However, one useful characteristic of linear programming problems is that, if an optimal solution exists and the solution space contains extreme points, there is always an optimal solution which is an extreme point of the solution space, or feasible region. Geometrically, this means an optimal solution always occurs at a vertex of the polyhedral solution space. In an $m$-dimensional feasible region, vertices occur where $m$ independent facets meet.

The integer hull of a feasible region is the convex hull of all the integer points in the region. If we could tighten the feasible region of the linear program to only its integer hull, so that every vertex is an integer solution, we could use linear programming methods to solve the LOP in polynomial time.

Therefore, if we knew enough about the geometry of $L^n$, the Simplex Method could be “tricked” into giving us an integer solution.

Interestingly, any work done on the LOP is also work done on the Traveling Salesperson Problem (TSP). Given a set of cities for a salesperson to visit, the TSP is to find the shortest trip between them such that the salesperson visits each city exactly once, and returns to the city they started in at the end of the trip. The LOP and the TSP are two very similar NP-complete

---

²The feasible region of a linear program is a polyhedron. An extreme point is a vertex of this polyhedron.
problems because in each problem the feasible solutions are orderings of a
finite set of elements.\cite{17}

Therefore any insight into the facets of $L^n$ may possibly be extended as
insight into the entire class of NP problems.

1.2 Motivation: Classes of Facets

Although the vertices of $L^n$ are simple to write down, the facets are not.
For general dimensions, a complete description of $L^n$ in terms of facets is un-
known. When new facets are discovered, they are discovered as new “classes”
of facets, with rules about how the facets look in general dimensions.

The facets of $L^n$ have been a topic of study since Grötschel, Jünger, and
Reinelt wrote about the “Facets of the Linear Ordering Polytope” in 1985.
They detailed four specific classes of facets of $L^n$.\cite{5}

In 1993, Reinelt continued by writing “A Note on Small Linear Ordering
Problems,” connecting the LOP to $L^n$, and listing the facets for $L^n$ up to
$n = 7$. In the paper, he finds that the minimum possible number of facets
for $L^7$ is 87,472, but does not prove it is the maximum. \cite{13}

In 1999, Bolotashvili et al. published “New Facets of the Linear Order-
ing Polytope,” adding to the known classes of $L^n$ facets using a rotational
method.\cite{2}

Although it may never be possible to find every class of facets of $L^n$, as
Reinelt is quoted as saying at the top of this chapter, the more classes which are known, the faster computations about the LOP and related NP-Hard problems will be. In this thesis, we search for potential new classes of facets of $L^n$.

1.3 Remarks

This thesis is partially an expository thesis, and partially a thesis of original work. We begin with expository work on polyhedral theory, optimization, and voting theory, then continue with a chapter of original work connecting $L^n$ with the voting problem.
Chapter 2

An Introduction to Polyhedral Theory

This chapter gives introductory definitions and theorems used in polyhedral theory in order to prepare for a closer examination of the linear ordering polytope. Much of the material provided here can be found in Chapter I.4 of Integer and Combinatorial Optimization by George Nemhauser and Laurence Wolsey [11] and in Chapter 2 of Introduction to Linear Optimization by Bertsimas and Tsitsikis [1].

Combinations and Hulls in $\mathbb{R}^n$

In this section we will be working with a finite set of vectors $\{x_0, \ldots, x_k \mid x_i \in \mathbb{R}^n\}$, and a corresponding set of scalars $\{\lambda_1, \ldots, \lambda_k \mid \lambda_i \in \mathbb{R}\}$. 

11
A linear combination is:

\[ \sum_{i=0}^{k} \lambda_i x_i = \lambda_0 x_0 + \lambda_1 x_1 + \ldots + \lambda_k x_k. \]

A ‘linear hull’ is the set of linear combinations for every choice of \( \{\lambda_1, \ldots, \lambda_k\} \).

This is most commonly referred to as the vectors’ span. See Figure 2.1 for the span of two linear independent vectors in \( \mathbb{R}^2 \).

Figure 2.1: In \( \mathbb{R}^2 \), for the vectors \((1, 1), (2, 3)\), the linear hull is all of \( \mathbb{R}^2 \).

A set of vectors is linearly independent if no vector in the set can be written as the linear combination of the other vectors in the set. Otherwise the set of vectors is called linearly dependent.

An affine combination of a finite set of vectors \( \{x_0, \ldots, x_k\} \) in \( \mathbb{R}^n \) is a linear combination, \( \sum_{i=0}^{k} \lambda_i x_i \), with the added constraint that \( \sum_{i=0}^{k} \lambda_i = 1 \).

The affine hull is the set of linear combinations for any choice of \( \lambda \) as
long as $\sum_{i=0}^{k} \lambda_i = 1$. See Figure 2.2 for an example of an affine hull of two vectors in $\mathbb{R}^2$.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig2_2}
\caption{In $\mathbb{R}^2$, for the vectors $(1, 1), (2, 3)$, the affine hull is the line connecting the points.}
\end{figure}

A set of vectors is \textit{affinely independent} if no vector in the set can be written as an affine combination of the other vectors in the set.

Two equivalent definitions of affine independence are:

For a set of vectors, $S = \{x^1, x^2, \ldots, x^k\} \subseteq \mathbb{R}^n$, are affinely independent if the set $S' = \{(x, -1) \mid x \in S\}$, where $(x, -1)$ is a vector in $\mathbb{R}^{n+1}$ resulting from appending an additional component containing $-1$ to the end of every $x^i$, is linearly independent.

A set of vectors, $S = \{x^1, x^2, \ldots, x^n\} \subseteq \mathbb{R}^n$, is affinely independent if the set $S' = \{x^i - x^k \mid x^i \in S \setminus k\}$ is linearly independent for any choice of $k$. 
Example of Affine Independence  The set of points $S = \{(1, 1), (2, 3), (0, 0)\}$ is affinely independent. We could check that it is impossible to write any of these three vectors as an affine combination of the other three, but it is more practical to use one of the other two definitions.

Using the first definition:

$S' = \{(1,1,-1), (2,3,-1), (0,0,-1)\}$. We can create a matrix with the elements of $S'$ as column vectors. We can then row reduce, knowing that these vectors are linearly independent iff the matrix is of full rank.¹

\[
\begin{bmatrix}
1 & 2 & 0 \\
1 & 3 & 0 \\
-1 & -1 & -1
\end{bmatrix} \xrightarrow{\text{rref}} \begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

The matrix is of rank 3 and thus of full rank.

This means the three vectors in $S'$ are linearly independent, and therefore $S$ is affinely independent.

Using the second definition:

Let us subtract $(1,1)$ from the other two vectors in $S$, although any of the three would work equally well.

Then $S' = \{(1,2), (-1,-1)\}$.

¹Rank: The rank of a matrix is the number of leading 1s when the matrix is in reduced row echelon form. An $n \times m$ matrix is of full rank if its rank is $n$. 
Using row reduction again:

\[
\begin{bmatrix}
1 & -1 \\
2 & -1 \\
\end{bmatrix}
\xrightarrow{\text{ref}}
\begin{bmatrix}
1 & 0 \\
0 & 1 \\
\end{bmatrix}
\]

We see our matrix is of full rank. So the set of vectors in \( S' \) is linearly independent and thus the vectors in \( S \) are affinely independent.

A conic combination of vectors \( \{x_0, \ldots, x_k\} \) in \( \mathbb{R}^n \) is a linear combination, \( \sum_{i=0}^{k} \lambda_i x_i \), where \( 0 \leq \lambda_i \quad \forall i \in [0, k] \).

The conic hull is the set of all linear combinations for any choice of \( \lambda \) as long as \( \lambda \) is nonnegative. See Figure 2.3 for an example of a conic hull of two vectors in \( \mathbb{R}^2 \).

![Figure 2.3: In \( \mathbb{R}^2 \), for the vectors (1, 1), (2, 3), the conic hull is the cone between the lines from the origin through each vector (hence the name).](image)

A convex combination of vectors \( \{x_0, \ldots, x_k\} \) in \( \mathbb{R}^n \) is a linear combination,
\[ \sum_{i=0}^{k} \lambda_i x_i, \] which is both affine and conic. Thus,

\[ \sum_{i=1}^{k} \lambda_i x_i \quad \text{where} \quad \sum_{i=1}^{k} \lambda_i = 1 \quad \text{and} \quad \lambda_i \geq 0 \quad \forall i \in [0, k]. \]

A **convex hull** is the set of all convex combinations of the elements in the set for any choice of \{\lambda_1, ..., \lambda_k\}. See Figure 2.4 for an example of a convex hull of two vectors in \( \mathbb{R}^2 \).

![Graph](image)

**Figure 2.4**: In \( \mathbb{R}^2 \), for the vectors \((1, 1), (2, 3)\), the convex hull is the intersection of the affine hulls and the conic hulls above, which is the line connecting the two points.

The main points from this section are summarized in the table below:

<table>
<thead>
<tr>
<th>type</th>
<th>restrictions on ( \lambda )</th>
<th>visually with 2 independent vectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>linear</td>
<td>none</td>
<td>all of ( \mathbb{R}^2 )</td>
</tr>
<tr>
<td>affine</td>
<td>( \sum_{i=0}^{k} \lambda_i = 1 )</td>
<td>line connecting vectors</td>
</tr>
<tr>
<td>conic</td>
<td>( \lambda_i \geq 0 \quad \forall i )</td>
<td>cone between two vectors from origin</td>
</tr>
<tr>
<td>affine</td>
<td>both from above</td>
<td>line segment connecting two vectors</td>
</tr>
</tbody>
</table>
Halfspaces and Convex Sets

A convex set, $S \subseteq \mathbb{R}^n$ is a set of vertices in which the convex combination of any two points in the set is also in the set:

For any $x, y \in S$, $\lambda x + (1 - \lambda)y \in S \quad \forall \lambda \in [0, 1]$.

Visually, this means the line segment joining every pair of vertices in the $S$ is also in $S$.

In two dimensions, a square is convex but a five-pointed star shape is not. For example, if we draw a line segment connecting two adjacent tips of the star, we will have a line segment that is not contained in the star shape (see Figure 2.5.) In contrast, it is impossible to connect any two points in a circle with a line segment that extends outside of the circle.

![Figure 2.5: A line drawn between points A, E lies outside the star shape. Thus, the star shape is not convex.](image)

**Theorem 1.** The intersection of two convex sets is also a convex set.

**Proof.** Consider the intersection of two convex sets $A, B \subseteq \mathbb{R}^n$, represented as $(A \cap B)$. 
Since $x, y \in A$ and $x, y \in B$ by the definition of set intersection, and since $A, B$ are both convex, any convex combination of $x, y$ is also in $A, B$:

$$\lambda y + (1 - \lambda)z \in A$$

$$\lambda y + (1 - \lambda)z \in B$$

for $\lambda \in [0, 1]$. So any convex combination of $y, z$ is in $A \cap B$:

$$\lambda y + (1 - \lambda)z \in (A \cap B).$$

Thus the intersection of any two convex sets is also a convex set.

**Corollary 1.** The intersection of a finite number of convex sets is also a convex set.

**Proof.** By induction.

The base case for the intersection of two convex sets is shown in the proof above.

Assume, for some number of sets, $k$, that if $S^1, S^2, ..., S^k$ are all convex sets, then $S^1 \cap S^2 \cap ... \cap S^k$ is also convex.

Now consider the case where we have the intersection of $k + 1$ convex sets: $S^1 \cap S^2 \cap ... \cap S^k \cap S^{k+1}$.

Let $C = S^1 \cap S^2 \cap ... \cap S^k$. We know $C$ is convex by the inductive assumption.
Then $C \cap S^{k+1}$ has been reduced to the intersection of only two convex sets, which we know is also convex by the base case.

Therefore, by induction, we see the intersection of a finite number of convex sets is also convex. \qed

A \textit{halfspace} is the set of elements in $\mathbb{R}^n$ that satisfy a linear inequality. That is, for a vector where every component is a scalar $a \in \mathbb{R}^n$ and a scalar $c$, 
\[
\{ x \in \mathbb{R}^n \mid a \cdot x \leq c \}
\] is the halfspace defined by the linear inequality $a \cdot x \leq c$. We could also represent this inequality as $\sum_{i=1}^{n} a_i x_i \leq c$.

The set 
\[
\{ x \mid a \cdot x \leq c \cap a \cdot x \geq c \}
\] is the \textit{hyperplane} defined by $a \cdot x = c$, and the set 
\[
\{ x \mid a \cdot x \leq c \cup a \cdot x \geq c \}
\] is the entire space $\mathbb{R}^n$.

\textbf{Theorem 2.} A halfspace is a convex set.

\textit{Proof.} By contradiction.

Assume a halfspace, $H \subseteq \mathbb{R}^k$, defined by $\sum_{i=0}^{k} a_i x_i \leq c$, is not convex. Then $\exists$ some convex combination of $y, z \in H$ such that $\lambda y + (1 - \lambda) z \notin H$ for $\lambda \in [0, 1]$.

Since $y, z \in H$, $\sum_{i=0}^{k} a_i y_i \leq c$ and $\sum_{i=0}^{k} a_i z_i \leq c$. Then
\[
\lambda \sum_{i=0}^{k} b_i y_i + (1 - \lambda) \sum_{i=0}^{k} d_i z_i \leq \lambda c + (1 - \lambda)c = c
\]

So any convex combination of $y, z \in H$, providing a contradiction. \qed
Corollary 2. The intersection of a finite number of halfspaces is a convex set.

The proof follows directly from Corollary 1 and Theorem 2.

A polyhedron is defined as the intersection of a finite number of halfspaces. Thus, from Corollary 2 we see a polyhedron is always a convex set.

Let $a$ be a vector in $\mathbb{R}^k$ such that all components are scalars, $x$ be a vector of variables in $\mathbb{R}^k$, and $c$ be a scalar. Then if the inequality

$$a \cdot x \leq c$$

holds for every $x$ in the polyhedron, $P$, we call the inequality a valid inequality for $P$.

2.1 Extreme Points

An extreme point is a vertex of a polytope. Extreme points can also be defined in terms of convex combinations.

For a polytope, $P$, a point $x$ in $P$ is an extreme point of $P$ iff it cannot be written as a convex combination of two other points in $P$.

Geometrically, this means an extreme point is one for which it is impossible to construct a line segment which runs through $x$ where the end points are both in $P$. See Figure 2.7.
CHAPTER 2. POLYHEDRAL THEORY

Figure 2.6: On the left is a filled-in square embedded in the plane of the paper in two dimensions. Its extreme points are \{A, B, C, D\}. On the right is a cube embedded in three dimensions. Its extreme points are \{A, B, C, D, E, F, H, G\}. Point F is not visible in the image.

Extreme points can also be defined by halfspaces.

For a polyhedron, \(P\), a point \(a \in P\) is an extreme point of \(P\) if there exists a halfspace such that the entirety of \(P\) is contained in that halfspace, and there is exactly one point in \(P\) which holds at equality for the halfspace-defining inequality. That point, \(a\), is the extreme point.

Proof that these definitions of extreme points are equivalent can be found in *Introduction to Linear Programming*. [1][p. 50]

**Example** Let our polyhedron, \(P\), be the square in \(\mathbb{R}^2\) with vertices at \{(0, 0), (1, 0), (0, 1), (1, 1)\}.

One halfspace which contains the entire square is \(x + y \leq 2\).
The only point \( a \in P \) which is contained in \( x + y = 2 \) is the point \((1, 1)\).
Thus \((1, 1)\) is an extreme point of \( P \).

This example is illustrated in Figure 2.8.

Figure 2.7: The point \((\frac{1}{2}, \frac{1}{2})\) is not an extreme point because it is the convex combination \( = \frac{1}{2}(1, 0) + \frac{1}{2}(0, 1) = \left( \frac{1}{2}, \frac{1}{2} \right) \) of \((1, 0), (0, 1) \in P\).

Figure 2.8: The point \((1, 1)\) is an extreme point of the shaded polygon because it is the unique element in \( P \) which is valid at equality for \( x + y \leq 2 \).


2.2 Polyhedra

Polyhedra can be defined as the intersection of a finite number of halfspaces or as the convex hull of a finite number of extreme points.

We saw above that a polyhedron can be defined as the intersection of a finite number of halfspaces. An example of this is given below.

Example \( P = \{ x \in \mathbb{R}^2 \mid (-x_1 + x_2 \geq 0 \cap -x_1 + x_2 \leq 2 \cap x_1 \geq 0 \cap x_1 \geq 2) \}. \)

Figure 2.9: The polyhedron \( P = \{ v \mid v \in (-x_1 + x_2 \geq 0 \cap -x_1 + x_2 \leq 2 \cap x_1 \geq 0 \cap x_1 \geq 2) \}. \)

The grey region in Figure 2.9 is \( P \).
A bounded polyhedron is called a \textit{polytope}.

The polyhedron, \( P \), in Figure 2.9 is bounded. The polyhedron is bounded below by \( y \geq 0 \), above by \( y \leq 4 \), to the left by \( x \geq 0 \), and to the right by \( x \leq 4 \).

For bounded polyhedra, we can equivalently define a \textit{polytope} as the convex hull of a finite number of extreme points.

Proof that any polytope can be represented either as the intersection of a finite number of halfspaces or as the convex hull of extreme points can be found in Bertsimas’ \textit{Introduction to Linear Optimization}.\[1\][p. 68]

An example of this construction is given below.

\textbf{Example} Given the finite set of points \( \{(0, 0), (1, 2), (3, 0)\} \), we can construct a polytope by taking every possible convex combination of the three points. In this case, every point is a vertex of \( P \). However, an equivalent polytope is the one constructed from the convex hull of \( \{(0, 0), (1, 0), (1, 2), (3, 0)\} \). Since \( (1, 0) = \frac{2}{3} (0, 0) + \frac{1}{3} (3, 0) \), it is a convex combination of other vertices and is not needed in the description of \( P \). See Figure 2.10.

A polyhedron, \( P \), is of dimension \( k \) if there are at most \( k + 1 \) affinely independent points in \( P \). This is written as \( \text{dim}(P) = k \).

\textbf{Example} The line segment in \( \mathbb{R}^2 \) defined by \( P = \{ x \in \mathbb{R}^2 \mid x_1 \leq 3, x_1 \geq 0, x_2 = 1 \} \) is a 1-dimensional polytope, as seen in figure 2.11.
We know that lines are 1-dimensional, so we can hypothesize that we will be able to find a set with \(1 + 1 = 2\) affinely independent points, but none with more than 2.

If we choose any two points on the line segment for our set, we can apply definition (b) of affine independence trivially. Any single vector in the set subtracted from the other will leave us with a single vector in \(S'\). Since any single vector is linearly independent, any set of two vectors on the line segment will be affinely independent.

This shows the line segment is of at least dimension 1. We now need to bound the dimension from above, showing that it cannot be greater than 1.

We can bound the dimension from above using another definition of affine independence. This definition of affine independence is that a set of vectors are affinely independent if no vector in the set can be written as an affine
Consider again that we have a set, $S \subseteq \mathbb{R}^2$, containing any two arbitrary elements from the line segment. Then the affine hull, the set of all affine combinations of those two vectors, is the infinite line connecting the two. Thus, any other vector on the line segment must be an affine combination combination of the two in $S$. So any third vector added to $S$ will be affinely dependent of the other two. So the maximum number of affinely independent points on the line segment is 2.

Since we can find an affinely independent set containing 2 vectors, but cannot construct an affinely independent set containing more than 2 vectors, it must be true that there are no more than 2 affinely independent vectors in $P$.

Thus, we know the dimension of $P$ is $2 - 1 = 1$, as hypothesized.

The equality constraint matrix of $P$, written as $A_e$, is the set of constraints

![Figure 2.11: The one-dimensional polytope is defined by $P = \{x \mid x_1 \leq 3, x_1 \geq 0, x_2 \leq 1, x_2 \geq 1\}$ is the line segment joining $A, B$ in $\mathbb{R}^2$.](image)
defining $P$, written in matrix form, that are met with equality for every point in $P$.

In the example above, in Figure 2.11, the constraint $x_2 = 1$ is a constraint met at equality for every point in $P$. We could write

$$A_\infty = \begin{bmatrix} x_1 & x_2 & b \\ 0 & 1 & 1 \end{bmatrix}.$$}

Note that the equality constraints need not be given directly in the problem statement, as is the case for this example (we were given $x_2 \leq 1, x_2 \geq 1$). This can make it difficult to write down all equality constraints for complicated polyhedra.

**Theorem 3.** For $R \subseteq \mathbb{R}^n$, $n = \dim(P) + \text{rank}(A_\infty)$.

Written differently: every linearly independent equality constraint of $P$ increases the codimension of of $P$ by 1.

This theorem is proved in Nemhauser and Wolsey’s *Integer and Combinatorial Optimization*, and relies on the fact that there are exactly $k + 1$ affinely independent points in a $k-$dimensional space.[11][p. 87]

**Example Continued** Continuing the above example from Figure 2.11, we have $A_\infty = \begin{bmatrix} x_1 & x_2 & b \\ 0 & 1 & 1 \end{bmatrix}$. Thus rank($A_\infty$) $\geq 1$, and dim($\mathbb{R}^2$) = 2, so we
know

\[ \dim(\mathbb{R}^2) = \dim(P) + \text{rank}(A) \]
\[ 2 \geq \dim(P) + 1 \]
\[ 1 \geq \dim(P) \]

This gives us an alternate way to give an upper bound on the dimension of \( P \), which we know is 1.

### 2.3 Faces and Facets

Given a valid inequality for a polyhedron \( P \), a face of \( P \) is the set of points in \( P \) which hold at equality.

For a valid inequality \( \pi \cdot x \leq c \) of \( P \subseteq \mathbb{R}^n \), where \( \pi \in \mathbb{R}^n \) and every component is a constant, \( x \in \mathbb{R}^n \) and every component is a variable, and \( c \in \mathbb{R} \), the face is:

\[ F = \{ x \in P \mid \pi \cdot x = c \}. \]

**Notes:** Since an extreme point is a unique point which holds at equality for a valid inequality, we can also define an extreme point as a zero-dimensional face of \( P \).

It is possible for a face to be empty. This will happen for any valid inequality which does not intersect \( P \).
\begin{figure}
\centering
\begin{tikzpicture}
\draw[->] (-2,0) -- (4,0) node[anchor=north] {$x$};
\draw[->] (0,-2) -- (0,2) node[anchor=east] {$y$};
\draw (0,0) -- (3,0) node[below] {$B$};
\draw (0,0) -- (0,1) node[below left] {$A$};
\end{tikzpicture}
\caption{The one-dimensional line segment defined by \( P = \{ x \mid x_1 \leq 3, x_1 \geq 0, x_2 = 1 \} \) between points \( A, B \) in two-dimensional space. \( x_2 \leq 0 \) is a valid inequality for \( P \). Every point in \( P \) holds at equality for \( x_2 = 1 \). So the face defined by \( x_2 = 1 \) is all of \( P \).}
\end{figure}

It is also possible for a face to be all of \( P \). This can happen if \( P \) is not full dimensional, or has a codimension \( > 0 \). See the example in Figure 2.12.

A face of \( P \) is called a \emph{proper face} if it is a face that is neither \( P \) nor \( \emptyset \).

A face is itself a polyhedron. It is the polyhedron defined by the halfspaces of \( P \), plus one additional halfspace which is the face-defining valid inequality.

A \emph{facet} is a face whose dimension is one less than the dimension of the polyhedron. For face, \( F \), and polyhedron, \( P \), \( F \) is a facet iff \( \dim(P) = \dim(F) + 1 \).
2.4 An Extended Example: The Wheel

This section demonstrates methods used to find classes of facets of a general polytope by examining a class of polytopes that arise from wheel graphs. The problem is posed in Exercise I.4.3 of *Integer and Combinatorial Optimization* by George Nemhauser and Laurence Wolsey.\[11\]

It is challenging to find the facets of the wheel polytope, yet this problem is deemed simple in the field of combinatorial optimization. This is because it is, in fact, possible to write down every facet of the wheel polytope of general dimension. In combinatorial optimization, it is not possible to enumerate every facet of the polyhedral solution space of an NP-Hard integer programming problem. If it were, then it would be possible to solve the integer program in polynomial time. The linear ordering polytope is one such solution space.\[13\][p. 68] We would like to motivate some of the challenges of finding facets of a combinatorially complex polytope in general dimension using the wheel polytope.

The wheel polytope is based on the wheel graph, $W_n$. The wheel is defined by the set of vertices, $V$, and the set of edges, $E$:

$$V = \{V_0, ..., V_n\}$$

$$E = \{(v_0, v_i) \mid i = 1, ..., n\} \cup \{(v_i, v_{i+1}) \mid i = 1, ..., n-1\} \cup \{(v_n, v_1)\}.$$

If there are $n + 1$ vertices, then there are $2n$ edges. There is one central
vertex, $v_0$, which is connected to every other vertex. The rest of the vertices are connected to two others, in a way that each non-central vertex is adjacent to exactly two other non-central vertices.

On 0, 1, 2, and 3 nodes, the wheel is trivial (it does not look like a wheel!).

On four nodes, $n = 3$, the wheel is shown in Figure 2.13.

![Figure 2.13: The wheel where $n = 3$, with $2n = 6$ edges.](image)

Let us call the edges connecting $v_0$ and another vertex the ‘spoke’ edges and the others the ‘non-spoke’ edges.

Based on $W_n$, we define a polyhedron:

$$P = \{ x \in \mathbb{R}^{|E|} \mid \sum_{e \text{ contains } v} x_e = 2, \forall v \in V \text{ and } 0 \leq x_e \leq 1, \forall e \in E \}$$
Every point $x$ in $P$ is a vertex-representation of the wheel graph’s weighted edges. Every component of $x \in P$ is associated with a particular edge of $W_n$. A point is in $P$ if the sum of the weights of the edges, $x_e$ touching every vertex of $W_n$ is exactly 2, and every $x_e$ is in $[0, 1]$.

**Theorem 4.** The dimension of $P$ is $n - 1$.

*Proof.* In order to find $\dim(P)$, we want to bound the dimension of $P$ by an upper bound and a lower bound which are equal. Let us first find an upper bound on $\dim(P)$.

In general, when $n = (\text{total number of nodes} - 1)$, $W_n$ has $2n$ edges. Since there is a component in every point in $P$ for every edge in $W_n$, $x \in P$ is in $\mathbb{R}^{2n}$ space. This means the maximum dimension of $P$ is $2n$.

Consider the equations defining $P$. There is one unique equation corre-
sponding to each vertex, requiring the edge weights of all edges adjacent to
that vertex to sum to 2. Since there are \( n + 1 \) vertices in \( P \), there are \( n + 1 \)
equality constraints in \( P \).

We know that each linearly independent equality constraint increases the
codimension of \( P \) by 1. Therefore, we now know the maximum dimension of
\( P \) is \( 2n - (n + 1) = n - 1 \).

Consider the set of equality constraints in \( P \). Up to relabelling vertices,
they can be written as:

\[
\sum_{i=1}^{n} x_i = 2 \\
x_i + x_{n+i} + x_{n+i+1} = 2 \quad \text{for } i \in [0, n-1] \\
x_n + x_{2n} + x_{n+1} = 2.
\]

The first sum forces the edges surrounding the middle vertex, \( x_0 \) to sum
to 2, and the second and third force each non-central vertex to have adjacent
edges which sum to 2.

For the wheel where \( n = 3 \), on four vertices, this is written as the system of
equations:
For \( n = 4 \), this looks like:

\[
S_4 = \begin{bmatrix}
e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & e_7 & e_8 & b_i \\
1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 2 \\
1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 2 \\
0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 2 \\
0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 2 \\
0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 2
\end{bmatrix}
\]

We can always use the following process of elementary row operations to
row reduce the equality matrix:

1. Move row 1 to the bottom of the matrix

2. Add \((-1)\) of every other row to the new bottom row

3. Divide the bottom row by \(-2\)

4. Add \((-1)\) of the bottom row to the two rows above where \(e_n\) is not 0.

After following this process, we will always be left with \(n + 1\) leading
1s. Therefore the rank of the matrix is \(n + 1\), so we have \(n + 1\) linearly
independent equality constraints.
For $S_3$, this process looks like:

\[
\begin{bmatrix}
  e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & b_i \\
  1 & 1 & 1 & 0 & 0 & 0 & 2 \\
  1 & 0 & 0 & 1 & 1 & 0 & 2 \\
  0 & 1 & 0 & 0 & 1 & 1 & 2 \\
  0 & 0 & 1 & 1 & 0 & 1 & 2 \\
\end{bmatrix} \rightarrow \begin{bmatrix}
  e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & b_i \\
  1 & 0 & 0 & 1 & 1 & 0 & 2 \\
  0 & 1 & 0 & 0 & 1 & 1 & 2 \\
  0 & 0 & 1 & 1 & 0 & 1 & 2 \\
\end{bmatrix} \rightarrow \begin{bmatrix}
  e_1 & e_2 & e_3 & e_4 & e_5 & e_6 & b_i \\
  1 & 0 & 0 & 1 & 1 & 0 & 2 \\
  0 & 1 & 0 & 0 & 1 & 1 & 2 \\
  0 & 0 & 1 & 1 & 0 & 1 & 2 \\
  1 & 1 & 1 & 0 & 0 & 0 & 2 \\
\end{bmatrix}
\]

(2.2)

So we always have $n + 1$ linearly independent equality constraints, and thus $\max \dim(P) = 2n - (n + 1) = n - 1$.

Next we need to show that $\min \dim(P) = 2n - (n + 1) = n - 1$ as well. This can be done by showing there exist $(n - 1) + 1 = n$ affinely independent
For any \( n \), we can construct a set of points, \( x \in P \) as follows:

1. Choose two edges, \( e_j, e_k \) from the set \( \{e_1, ..., e_n\} \) such that \( k = j + 1 \) or \( j = 0 \) and \( k = n \) (Choose 2 edges going out from the spoke such that the edges are ‘next to’ each other).

2. Give those 2 edges a weight = 1 and the edges in \( \{e_1, ..., e_n\}\setminus\{e_j, e_k\} \) (the rest coming out of the spoke) a weight = 0.

3. Set the weight of \( e_{j+n} = 0 \) and the weight of every edge in the set \( \{e_{n+1}, ..., e_{2n}\}\setminus\{e_{j+n}\} = 1 \) (Set the weight of the edge ‘between’ those from the spoke with weight 1 to a weight of 0, and set the rest of the edges to a weight of 1).

4. There are \( n \) such points/graphs in \( P \). Let us call this the ‘Set of Pac-Man points,’ since if the edges are colored according to weight, they outline a Pac-Man shape. See Figure 2.16.

For \( n = 3 \), these 3 graphs are shown in Figure 2.15.

These graphs, written as vertices in \( \mathbb{R}^6 \), where each component is the weight of \( e_i \), are:

\[
\{(1, 1, 0, 0, 1, 1), (0, 1, 1, 1, 0, 1), (1, 0, 1, 1, 0, 1)\}.
\]

We will show this set of \( n \) elements in \( P \) is a set of \( n \) affinely independent
If we put these elements into matrix-form and row reduce, we get:

\[
\begin{bmatrix}
1 & 1 & 0 & 0 & 1 & 1 \\
0 & 1 & 1 & 1 & 0 & 1 \\
1 & 0 & 1 & 1 & 1 & 0 \\
\end{bmatrix} \rightarrow \begin{bmatrix}
1 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 & 0 & 0 \\
\end{bmatrix}
\]

In this example, we see the rank of the matrix is 3, and therefore we have
found 3 linearly independent points. Since linear independence implies affine independence, we have found 3 affinely independent points.

It will always be the case that a set of $n$ points constructed in this way will produce a matrix of full rank, so we have found $n$ affinely independent points in any $P$.

Since there are $n$ affinely independent points in $P$, $\min \dim(P) = (n-1)$.

Therefore, since the maximum dimension of $P$ is $(n-1)$ and the minimum dimension of $P$ is also $(n-1)$, we know $\dim(P) = (n-1)$.

\[ \Box \]

**Theorem 5.** The inequality constraints $x_e \geq 0$ are redundant.

**Proof.** For contradiction, assume there exists some edge such that $x_e < 0$ with weight $w$. 
We know that every edge touches at least one vertex with exactly 3 adjacent edges (these will be all non-central vertices). Call the remaining two edge weights $y, z$. Each of these edge weights must be less than or equal to 1 as defined in the problem.

Then the sum of the three edge weights is

$$ w + y + z, $$

where $w < 0, y \leq 1, z \leq 1$. So

$$ w + y + z < 0 + y + x \leq 0 + 1 + 1 = 2 $$

and

$$ w + y + z < 2. $$

However, we also have the constraint that the sum of the edges adjacent to any vertex $= 2$.

Thus we have reached a contradiction and it must not be possible to have an edge weight less than 0.

Since it is not possible to construct a graph with a negative edge weight that also follows the rest of the constraints, the set of constraints specifying $x_e \geq 0$ must be redundant.\hfill $\square$

**Theorem 6.** The inequality constraints $x_e \leq 1$ are redundant for $e = (v_0, v_i)$ for $i = 1, ..., n$ (the spoke edges).
Proof. Consider a general wheel polytope, $P$, with vertices $\{v_0, ..., v_n\}$. For contradiction, assume there exists some spoke $e = (v_0, v_k)$, such that its corresponding edge weight $> 1$. We will show this is not possible, given the rest of the constraints on $P$.

We will proceed with a combinatorial argument.

Consider the set of all vertices excluding $v_0$ and $v_k$. Call this set $S$. We know $|S| = (n - 1)$. Since there is a set of constraints requiring the sum of the weights of the edges adjacent to each vertex to be 2, the total sum of all edge weights adjacent to any vertex in $|S|$, taking double-counting into consideration, must be $2(n - 1)$.

We can split the edges which are adjacent to $|S|$ into three classes:

1. The spokes, excluding the one connecting $(v_0, v_k)$. There are $(n - 1)$ of these. Each is adjacent to a single vertex in $S$ and thus contributes to our sum $2(n - 1)$ once.

2. The non-spokes adjacent to $v_k$. There are exactly two of these. Each is adjacent to a single vertex in $S$ and thus contributes to our sum $2(n - 1)$ once.

3. The rest of the non-spokes. There are $(n - 2)$ of these. Each is adjacent to two vertices in $S$ and therefore contributes to our sum $2(n - 1)$ twice.

In total, for a valid wheel polytope, it must be true that the sum of the total weights of all three classes must be exactly $2(n - 1)$. Let us examine
the sum of these classes.

Since we have one spoke $e = (v_0, v_k)$ with edge weight $> 1$, and every spoke is adjacent to $v_0$, the sum of the edge weights of the rest of the spokes must be $< 1$. Thus the total sum of class (1) must be $< 1$.

Similarly, the total sum of class (2) must also be $< 1$.

Finally, since class (3) contains only non-spokes, and we have a constraint specifying that the edge weight of every non-spoke must be $\leq 1$, the sum of these edges is at most $(n - 1)$. Since each edge is adjacent to two vertices in $|S|$, and therefore must be counted twice, the total sum of class (3) must be $\leq 2(n - 2)$.

Therefore, the total sum of all three classes is $< 1 + 1 + 2(n - 2) = 1 + 1 + 2n - 4 = 2n - 2 = 2(n - 1)$.

So we see the sum of the three classes, which must be exactly $2(n - 1)$ for a valid wheel polytope, is strictly less than $2(n - 1)$ in this example.

Therefore, it must not be possible for any spoke to have an edge weight greater than 1 provided the rest of the constraints.

Therefore the inequality constraints $x_e \leq 1$ are redundant for $e = (v_0, v_i)$ for $i = 1, ..., n$.

\[\square\]

**Theorem 7.** A minimal representation of $P$ is \{ \(\sum_{i \in \{1, ..., n\}} c_i = 2\),

\[x_i + x_{n+i} + x_{n+i+1} = 2 \text{ for } i \in [0, n - 1],\]
\[ x_n + x_{2n} + x_{n+1} = 2, \ e_i \leq 1 \ for \ i \in [n + 1, 2n] \}.

Proof. Since all \( x_e \geq 0 \) and \( x_e \leq 1 \) for \( e \) connecting \( v_0 \) to \( i \in \{1, ..., n\} \) are redundant, they are not necessary in the representation of \( P \) by equality and inequality constraints. Therefore a representation of \( P \) that is equivalent to the one given at top is:

\[
\{ \sum_{i \in \{1, ..., n\}} e_i = 2, \\
x_i + x_{n+i} + x_{n+i+1} = 2 \ for \ i \in [0, n - 1], \\
x_n + x_{2n} + x_{n+1} = 2, \ e_i \leq 1 \ for \ i \in [n + 1, 2n] \}
\]

We will show this is a minimal representation by showing that the removal of any constraint allows the construction of a point that is not in \( P \).

Let us consider the constraints in three classes:

1. \( \sum_{i \in \{1, ..., n\}} e_i = 2 \) (the constraint which specifies the edges surrounding the center vertex sum to 2)
2. \( x_i + x_{n+i} + x_{n+i+1} = 2 \ for \ i \in [0, n - 1] \) (the set of constraints requiring the 3 edges adjacent to a non-center vertex to sum to 2)
3. \( e_i \leq 1 \ for i \in [n + 1, 2n] \) (the set of constraints specifying the weight of a non-spoke edge is less than or equal to 1)

It is sufficient to find a single counter-example for each class.

1. A counter-example for \( \sum_{i \in \{1, ..., n\}} e_i = 2 \) is the graph in \( n = 3 \) with edge weights = \( (0, 0, 0, 1, 1, 1) \). See Figure 2.17. Since this graph is not \( \in P \)
for \( n = 3 \) but is allowed with the reduced constraint set, it must be that \( \sum_{i \in \{1, \ldots, n\}} e_i = 2 \) is a necessary constraint for \( P \).

2. A counter-example for \( x_i + x_{n+i} + x_{n+i+1} = 2 \) for \( i \in [0, n - 1] \) is the graph in \( n = 3 \) with edge weights = \( (1, 1, 0, 1, 0, 0) \). See Figure 2.18.

3. A counter-example for \( e_i \leq 1 \) for \( i \in [n + 1, 2n] \) is the graph in \( n = 4 \) with the edge weights = \( (0, 0, 1, 1, 2, 0, 1, 0) \). See Figure 2.18.

Since it is possible to find a graph that is not in \( P \) by removing any single constraint in the representation above, that representation is minimal.

Figure 2.17: Counterexample for (1). If we do not require the sum of the edges touching the center vertex to be 2, this graph is allowed. But \( (0, 0, 0, 1, 1, 1) \notin P \).
CHAPTER 2. POLYHEDRAL THEORY

Figure 2.18: Counterexample for (2). If we do not require the sum of the adjacent edges touching some vertex to be 2 (in this case, the upper right vertex), this graph is allowed. But \((1, 1, 0, 1, 0, 0) \notin P\).

Figure 2.19: Counterexample for (3). If we do not require all non-spoke edges to have a weight less than or equal to 1, this graph is allowed. But \((0, 0, 1, 1, 0, 1, 0, 2) \notin P\) for \(n = 4\).
Chapter 3

The Linear Ordering Polytope

We introduced the linear ordering polytope, $L^n$, in Chapter 1, but we will expand upon its features here and give a few specific examples to help the reader understand the object.

A permutation is a function, $\pi : S \to S$ which is both one-to-one and onto (bijective). For a set with $n$ elements, there are $n!$ permutations.

Each permutation corresponds to a permutation vector, $x \in \mathbb{R}^{2(n)}$, as follows. For each component of $x$:

$$x_{ij} = \begin{cases} 1, & \text{if } \pi(i) > \pi(j). \\ 0, & \text{otherwise.} \end{cases}$$

This gives us a set of $n!$ vectors in $\mathbb{R}^{2(n)}$. 

45
CHAPTER 3. THE LINEAR ORDERING POLYTOPE

Given a set, \( S \), with \( n \) elements, \( L^n \) is the convex hull of the set of all permutation vectors of \( S \).

For example, if \( S = \{A, B, C\} \), we have the \( n! = 6 \) permutations which result in an ordering:

\[ \{ABC, ACB, BAC, BCA, CAB, CBA\} \]

If we let the components of \( \mathbb{R}^6 \) be in the order

\[ (AB, BA, AC, CA, BC, CB), \]

these correspond to the set of permutation vectors

\[ \{(1, 0, 1, 0, 1, 0), (1, 0, 1, 0, 0, 1), (0, 1, 1, 0, 1, 0)\} \]

\[ (0, 1, 0, 1, 1, 0), (1, 0, 0, 1, 0, 1), (0, 1, 0, 1, 0, 1)\} \]

The convex hull of these vectors is \( L^3 \).

**Theorem 8.** Every permutation vector is a vertex of \( L^n \).

**Proof.** For contradiction, assume there exists a permutation vector that is not a vertex of \( L^n \). Without loss of generality, let this vector be labelled \( x^{n!} \). Then there must be some convex combination of the other \( (n! - 1) \) permutation vectors of \( L^n \) that equals \( x^{n!} \).

Let the rest of the permutation vectors be labelled \( \{x^1, x^2, ..., x^{n!-1}\} \). Then
for scalars \( \{\lambda^1, \lambda^2, ..., \lambda^{n!-1}\} \) where \( \lambda^i \in [0, 1] \):

\[
\sum_{i=1}^{n!-1} \lambda^i x^i = x^{n!} \quad \text{where} \quad \sum_{i=1}^{n!-1} \lambda^i = 1.
\]

By definition, every permutation vector is a binary vector thus each component is either 0 or 1.

Case 1: Consider the situation where a component of \( x^{n!} \) is 1. Let this component be labelled \( x^{n!}_j \). Then in the convex combination, the sum of the \( j \)th components must be:

\[
x^{n!}_j = \sum_{i=1}^{n!-1} \lambda^i x^i_j = 1.
\]

Since \( x_j \leq 1 \) because the permutation vectors are binary, \( \sum_{i=1}^{n!-1} \lambda^i = 1 \), and all \( \lambda \)s are nonnegative, \( \sum_{i=1}^{n!-1} \lambda^i x^i_j = 1 \) only if \( x^i_j = 1 \) for every \( i \in \{1, ..., n! - 1\} \).

Thus the \( j \)th component of every \( x^i \) must equal 1.

Case 2: Similarly, consider the situation where a component of \( x^{n!} \) is 0. Let this component be labelled \( x^{n!}_j \).

Using the same reasoning as in case 1, \( x^{n!}_j = \sum_{i=1}^{n!-1} \lambda^i x^i_j = 0 \) only if the \( j \)th component of every \( x^i \) is equal to 0.

Combining cases 1 and 2, the only way for \( x^{n!} \) to be the result of a convex combination of the other \( n! - 1 \) permutation vectors is if every other
permutation vector is exactly \( x^n \). But every permutation vector is unique, and we assumed \( x^n \) was not a vertex.

Therefore, it must be true that \( x^n \) is a vertex of \( L^n \), and we see that every permutation vector is a vertex of \( L^n \).

From these definitions of \( L^n \), we notice that not all \( 2\binom{n}{2} \) components are necessary to describe \( L^n \). This is because if \( x_{ij} = 1 \) then \( x_{ji} = 0 \) and vice versa. Therefore we can fully describe \( L^n \) in \( \binom{n}{2} \)-space, using only one component of each \( x_{ij} \) and \( x_{ji} \) pairing. Let us call this depiction of \( L^n \) the standard depiction.

We have a choice of which components to include in our description; for all \( \binom{n}{2} \) pairs of elements of \( S \), we may choose either \( x_{ij} \) or \( x_{ji} \). So we have \( 2\binom{n}{2} \) possibilities for components, and \( 2\binom{n}{2} \) distinct representations of \( L^n \) in the standard depiction up to reordering of the components.

Despite the fact that our choice of axes determines the permutation vectors in \( \binom{n}{2} \)-space, each of the standard representations of \( L^n \) are equivalent. This is because \( L^n \) is a \( \binom{n}{2} \)-dimensional polytope embedded in \( 2\binom{n}{2} \)-space. We arrive at any chosen standard depiction of \( L^n \) by removing dimensions related to equality constraints of \( L^n \). These equality constraints are all \( \binom{n}{2} \) constraints which state \( x_{ij} + x_{ji} = 1 \) for every pair of elements, or if \( x_{ij} = 1 \), then \( x_{ji} = 0 \).

Our choice of which coordinate of \( L^n \) to remove dictates the vector-representation of the standard description of \( L^n \), but it is still the same
polytope revealed by removing dimensions from the space it is embedded in.

After removing the \( \binom{n}{2} \) equality constraints, we are left with the standard depiction in \( \binom{n}{2} \) dimensions. We tend to use the standard depiction to study \( L^n \) because it is full-dimensional in its space.

The standard depiction of \( L^3 \), where the cardinality of \( S = 3 \), is shown in Figure 5.1. \( L^3 \) includes six of the eight vertices of the unit cube. The two vertices not included represent cycles, not permutations. For example, if we use the three components \( (AB, AC, BC) \), then the vertex \((0,1,0)\) would not be included as it says \( B \) come before \( A \), \( A \) comes before \( C \) and \( C \) comes before \( B \). This is not a valid permutation.

The point \((1,0,1)\) would also be excluded from \( L^3 \) by similar reasoning.
Figure 3.1: The three-dimensional $L^n$. It is a 3-cube with an equilateral pyramid sliced out of two opposite corners. One of those corners is shown at the front; the other is not visible at the back.
Chapter 4

Linear Programming, Integer Programming, and the Linear Ordering Polytope

In this chapter, we introduce a variety of topics which are necessary to understand the connections between voting tournaments and the linear ordering polytope which will be established in Chapter 5. These topics include acyclic tournaments, two optimization problems related to $L^n$, and linear programs.

4.1 Acyclic Tournaments and $L^n$

This section discusses acyclic tournaments:
Definition. A tournament is a complete directed graph where, if $i, j$ are nodes of the graph, either edge $e_{ij}$ or $e_{ji}$ are in the graph, but not both. An acyclic tournament is a tournament without any cycles.

An acyclic tournament with $n$ vertices can be fully described by a permutation, $\pi$ of those $n$ elements.

In the permutation, $\pi(i) > \pi(j)$ iff the node labelled $j$ has a directed arrow toward the node $j$ in the tournament. We often think about permutations as an ordered list of elements. In this form, $i$ would be to the left of $j$ in the list.

Only acyclic tournaments can be represented by permutations because both permutations and acyclic tournaments preserve transitivity and cyclic tournaments do not. That is, if $A > B$ and $B > C$, then $A > C$ for all elements in the set.

Since tournaments can be completely expressed through permutations, they can also be represented as $L^n$. 
4.2 The Linear Ordering Problem

As defined in Chapter 3, a permutation is a bijective function \( \pi : S \rightarrow S \). We say \( x_i \) comes ‘before’ \( x_j \) in an ordering, written as \( x_i > x_j \), if \( \pi(i) > \pi(j) \). For a set of size \( n \), there are \( n! \) permutation functions.

We can assign a value for all \( 2^{(n)} \) ordered pairs \( (i, j) \), \( w_{ij} \).

Then every permutation, \( \pi^k \), has a value,

\[
    c_{\pi^k} := \sum_{i \neq j} p_{ij}^k w_{ij}
\]

where component \( p_{ij}^k \) is a presence/absence component such that \( p_{ij}^k = 1 \) if \( i \) comes before \( j \) in the permutation and \( p_{ij}^k = 0 \) otherwise.

Let \( S' \) be the set of all possible orderings. Then the Linear Ordering Problem (LOP) is to select the permutation, \( k \), which maximizes \( c_{\pi^k} \):

\[
    \max c_{\pi^k}
\]

subject to \( \pi^k \in S' \)

If we were to use the linear ordering polytope to visualize the LOP, the problem would be to find the vertex of \( L^n \) with the maximum \( c_{\pi^k} \).

The LOP is much more compact to write down that the TSP, however the problems are closely related, as we will see in the next section.
4.3 The Travelling Salesperson Problem

“Christos Papadimitriou told me [John Bently] that the travelling salesman problem is not a problem, it’s an addiction.”[3, p. 211]

The Travelling Salesperson Problem (TSP) is a long-studied problem in complexity theory. The TSP has ended up at the heart of the $P$ versus $NP$ debate: does the class $P = NP$? Meaning, if we have a problem that has a polynomial-time verifier (in $NP$), does it always also have a polynomial-time solver (in $P$)?[16] Since the TSP is $NP$–complete, if an efficient, polynomial-time algorithm can be written for the TSP, then an efficient algorithm can be written for every problem in $NP$.[3, p. 9]

Given a set of cities that must be visited by a salesperson and the distances between each pair of them, the TSP is to find the minimum distance a salesperson must travel in order to visit the entire set of cities exactly once, then return to the city they started in.

Let the set of $n$ cities be $\{c_1, c_2, ..., c_n\}$, and the weights between them represented by $w_{ij}$ where $w_{ij}$ is the distance between city $i$ and city $j$. Also let $\pi$ be a binary vector in $\mathbb{R}^{2^\binom{n}{2}}$ where $\pi_{ij} = 1$ if the salesperson travels from city $i$ to city $j$ in the solution, or $\pi_{ij} = 0$ otherwise.

Then the TSP can be written as an integer program, but often the problem is solved using the “degree relaxation LP”[3, p 110]:
\[
\begin{align*}
\min & \quad \sum_{i \neq j} \pi_{ij}w_{ij} \\
\text{subject to} & \quad \sum_{i \neq j} \pi_{ij}^p + \pi_{ji}^p = 2 \\
& \quad \pi_{ij} \in \{0, 1\}
\end{align*}
\]

The objective function minimizes the total distance travelled, the first set of constraints requires the city to be entered exactly once and exited exactly once when combined with the second set of constraints requiring the components of \( \pi \) to be either 0 or 1.

At first glance, this seems like the correct IP for the TSP. However, it is possible for this IP to result in disjoint cycles rather than one cycle connecting the entire set of cities (see Figure 4.1). Thus, more constraints are needed.

The constraints which are needed are called the subtour elimination constraints. These are constraints which force some subset of the cities to be connected to the rest with at least one incoming edge and one outgoing edge.[3, p 113]

For our example in Figure 4.1, let the left cycle be \( X \) and the right be \( Y \). Then a pair of subtour elimination constraints would be:

\[
\sum_{i \in X, j \in Y} \pi_{ij} \geq 1 \quad \sum_{i \in X, j \in Y} \pi_{ji} \geq 1
\]

The first requires that at least one edge leaves the left cycle and goes to
Figure 4.1: An example of a valid solution according to the IP above, but it is not a valid TSP solution because it contains two disjoint cycles.

the right one, and the second is the opposite.

There are many subtour constraints required to give a full representation of the TSP as an IP. Thus, when solving the TSP for large $n$, some mathematicians will add subtour constraints in small batches until a cycle connecting all cities emerges as an optimal solution. Since adding constraints reduces the feasible region, adding constraints results in an optimal value which is worse than or the same as the original LP or IP. Therefore as soon as a cycle emerges from the relaxed TSP IP, we know it is an upper bound for the objective.

A seen in Figure 4.1, the TSP is an integer program that is hard to write down, and hard to solve. Any connections to other areas of mathematics could possibly help with finding better general solutions to the TSP.
CHAPTER 4. LP, IP, AND $L^N$

Theorem 9. The LOP can be reduced to the TSP.

Proof. Take a finite set of elements to be used in the LOP, $S = \{c_1, c_2, ..., c_n\}$.

Add a dummy city to $S$, $d$, such that the weight from any $c_j$ to $d$ is the same, or $w_{id} = w_{jd} \ \forall i, j \in S$, and such that $w_{id}$ is greater than any other $w_{ij}$ from the original set.

Then the optimal LOP permutation of $S$ is the same as the optimal TSP cycle in $S \cup d$, beginning the cycle with the element after $d$ and removing $d$.

This gives a construction of a reduction from LOP to TSP. \qed

Example Consider the problem with four elements, where $S = \{c_1, c_2, c_3, c_4\}$, and weights are as shown in Figure 4.2.

![Figure 4.2: An example of the LOP to TSP reduction. On the left is the set of cities and weights before that we would like the optimal LOP permutation. On the right is the set after adding the dummy city. If the TSP is solved on this set, it also solves the LOP.](image-url)
Since the lowest $w_{ij} = 5$, we can add a dummy city, $d$, such that $w_{id} = 6$ for every $i$. Any cycle in $S \cup d$ must contain the weight $w_{id} = 6$ twice, regardless of which two $c_i$ it is adjacent to in the cycle. Removing that choice, the remaining decision is to choose the lowest-weight permutation of $S$, which is the LOP for $S$. 


4.4 Linear Programming Basics

This section reminds readers of some introductory definitions and terms for linear programming. It follows from my studies of Bertsimas and Tsitsiklis’ Introduction to Linear Optimization [1].

An example of a linear program:

\[
\begin{align*}
\text{min} & \quad -2x_1 - 3x_2 + \alpha x_3 \quad \text{(objective function)} \\
\text{subject to} & \quad (\text{three constraints}) \\
& \quad 2x_1 + 2x_2 + x_3 \leq 60 \\
& \quad 2x_1 - x_2 + 5x_3 \leq 20 \\
& \quad 5x_1 - 11x_2 + x_3 \leq 15 \\
\text{(nongativity constraints)} & \quad x_1, x_2, x_3 \geq 0
\end{align*}
\]

A linear program (LP) consists of a set of variables (written as a vector, \( \mathbf{v} \)), an objective function, and a set of constraints. The objective function must be of the form “maximize/minimize \( \pi \cdot \mathbf{v} \), where \( \pi \) is a set of real-valued coefficients for the variables. The set of constraints is of the form \( \pi_j \cdot \mathbf{v} \leq x \) where \( x \) is a real number.

A feasible solution to a linear program is a set of values for \( \mathbf{v} \) that satisfies all constraints. The feasible region for an LP is the set of all feasible solutions. Because all constraints are linear and therefore define a halfspace, the feasible region is a convex polyhedra defined by a finite number of halfspaces.
A basic solution to a linear program with \( n \) variables is a set of values for \( \mathbf{v} \) for which there are \( n \) linearly independent constraints of the LP which are satisfied at equality.

A basic feasible solution (BFS) to a linear program is one that is both basic and feasible. These solutions map to the “corner points” or “extreme points” of the feasible region.

The feasible region is also the convex hull of the set of BFS.

**Theorem 10.** An optimal solution to a LP exists if and only if one of the BFS is an optimal solution.

This theorem is proved as Theorem 2.7 in [1][p. 65].

Every LP has a dual, which is another LP constructed from the original (the “primal”) according to a specific set of instructions outlined below.

<table>
<thead>
<tr>
<th>If Primal is:</th>
<th>Then Dual is:</th>
</tr>
</thead>
<tbody>
<tr>
<td>max objective function</td>
<td>min objective function</td>
</tr>
<tr>
<td>coefficient matrix ( \mathbf{A} )</td>
<td>coefficient matrix ( \mathbf{A}^T )</td>
</tr>
<tr>
<td>objective coefficient vector = ( \mathbf{c} )</td>
<td>RH side constraint vector = ( \mathbf{c}^T )</td>
</tr>
<tr>
<td>RH side constraint vector = ( \mathbf{b} )</td>
<td>objective coefficient vector = ( \mathbf{b}^T )</td>
</tr>
<tr>
<td>variable ( x \geq 0 )</td>
<td>corresponding constraint ( \leq 0 )</td>
</tr>
<tr>
<td>variable ( x \leq 0 )</td>
<td>corresponding constraint ( \geq 0 )</td>
</tr>
<tr>
<td>variable ( x ) free</td>
<td>corresponding constraint = 0</td>
</tr>
</tbody>
</table>

In Chapter 5, we will use the dual of a LP to make connections between
voting and the facets of $L^n$.

**Theorem 11.** The dual of the dual is the primal.

This theorem is proved in [1]. It tells us that if we take the dual of some LP twice, the result is that exact LP.

An example of a linear program (Let this be the primal):

\[
\begin{align*}
\text{max} \quad & -5x_1 - 15x_2 + 6x_3 + x_4 \\
\text{subject to} \quad & x_1 + 2x_2 + 5x_3 \leq 60 \\
& -2x_1 + x_2 + 2x_3 + 2x_4 = 20 \\
& 2x_1 + 2x_2 + x_3 + x_4 \geq 15 \\
& x_2, x_4 \geq 0, x_3 \text{ free, } x_1 \leq 0
\end{align*}
\]

Then its dual is:

\[
\begin{align*}
\text{min} \quad & 60y_1 + 20y_2 + 15y_3 \\
\text{subject to} \quad & y_1 - 2y_2 + 2y_3 \geq -5 \\
& 2y_1 + y_2 + 2y_3 \leq -15 \\
& 5y_1 + 2y_2 + y_3 = 6 \\
& 2y_2 + y_3 \leq 1 \\
& y_1 \geq 0, y_2 \text{ free, } y_3 \leq 0
\end{align*}
\]

**Theorem 12.** Strong Duality Theorem: If the primal has a optimal value,
$k \in \mathbb{R}$, then the dual has an optimal value, and that value is $k$.

The Strong Duality Theorem is proved in [1][p. 148]. This theorem tells us that if there is an optimal solution, it always occurs at one of the basic feasible solutions, or one of the vertices of the feasible region.

It is possible for multiple, even an infinite number, of points in the feasible region to be optimal solutions, but one of those solutions will always be a BFS.

**Complexity Theory for Linear Programming**

Linear programs are formally efficient to solve. For example, the ellipsoid algorithm can be used to solve a LP in polynomial-time, meaning LPs are in the class $P$.[11] In practice many linear programming software tools use the Simplex Algorithm. Simplex has not been proved to be in $P$, but in practice it performs faster than most other linear program solvers.[9]

**4.5 Integer Programming from Linear Programming**

The voting tournament that is central to our connection between $L^n$ and voting is an integer program. In this section, we use what we know about linear programming to develop ideas on integer programming.
An integer program (IP) follows the same rules as a linear program, except variable solutions are restricted to integer (Z) values, rather than any real value (R). These solutions containing only integer values are the lattice points in the feasible region of the linear program.

**Definition.** A lattice point in \( \mathbb{R}^n \) is a point in \( \mathbb{R}^n \) where every component is restricted to integer values.

Integer programs are much harder to solve than linear programs. There is no polynomial-time algorithm to solve integer programs, and in fact, integer programming is NP-complete.[12, p 202]. However, under certain circumstances, it is possible to use linear programming algorithms to solve integer programming problems.

**Example** Consider the IP:

\[
\begin{align*}
\text{max} & \quad y \\
\text{subject to} & \quad 0.8x - y \leq 0 \\
& \quad -0.8x - y \leq -4 \\
& \quad x, y \geq 0 \\
& \quad x, y \in \mathbb{Z}
\end{align*}
\]

The feasible region is shown in Figure 4.5.

If we were to solve this as a linear program, i.e. excluding the constraint
$x, y \in \mathbb{Z}$, the maximum value of $y$ in the feasible region is 2, so the optimal value is 2.

However, this value only occurs at the point (2.5, 2) in the feasible region. Thus, this is not a solution to the IP. A solution to the IP must occur at a lattice point.

By looking at the image, we can see the optimal value is 1 for the IP. There are two lattice points where this value occurs, (2, 1) and (3, 1).

Notice the optimal value of the IP does not occur at a vertex of the feasible region! This is part of what makes solving IPs so difficult, since solving LPs in polynomial time relies deeply on Theorem 10: If the feasible region has vertices, then every vertex is a BFS, and at least one of them must have the optimal value.
One way around this problem is by reducing the feasible region to be the convex hull of the lattice points in the feasible region.

**Theorem 13.** If we can write the reduced feasible region for an IP, where the reduced feasible region is the convex hull of the lattice points of the feasible region, then we can solve an IP as a LP in polynomial time.

The convex hull of the lattice points of our example feasible region is shown in Figure 4.5.

After finding the reduced feasible region, the optimal solution to the IP is now a corner point of the new region, so we can solve the IP as an LP and find the optimal solution.
The Voting LP and its Dual

This thesis explores the connections between a specific optimization problem in voting theory, as outlined in the introduction, and the facets of $L^n$. This section introduces the specific linear program used to formulate the connections.

The linear program is based on a problem which uses a ranked voting system. This system has individuals submit ballots which list every candidate in order of preference. The set of all possible ballots is the set of all possible permutations of the set of candidates.

A tournament graph $T = (V, E)$ is a complete directed graph. For a set of $n$ candidates, there are $n$ nodes in the tournament graph, one representing each candidate. There are $\binom{n}{2}$ edges, one directed edge between each pair of candidates. An edge is directed toward $i$ from $j$ if a majority of ballots prefer $i$ to $j$.

Given a collection of ballots and an edge $ij \in E$, we define the edge agreement to be the fraction of the ballots for which $i$ is preferred to $j$.

The linear program for a tournament $T = (V, E)$ finds the maximum $\alpha$ for which there exists a collection of ballots so that the edge agreement is at least $\alpha$ for every edge in $E$.

Then the optimal value of the LP for a tournament $T$ is called the agreement for tournament $T$. (Aside: $\frac{1}{2} < \text{agreement} \leq 1$).
As an example, suppose in an election with candidates \{A, B, C, D\}, we are given four ballots \((ABCD), (ABDC), (ACBD), (ADBC)\). Then the edge agreements are as follows:

In the above example, the overall agreement would be \(\frac{1}{2}\), the minimum edge agreement.

This tournament is shown in figure 4.6.

We know, by construction, that the tournament shown in figure 4.6 can
be created with $\frac{1}{2}$ agreement. But is this the maximum agreement that can be used to construct the graph?

The answer is no. Since the tournament in figure 4.6 is acyclic, it can be constructed using a single permutation and therefore unanimous agreement, or agreement = 1. This single ballot is $(ABCD)$.

A linear program was constructed by Shepardson and Tovey in [15] to represent this problem. In some sense, linear program searches through all possible ballot combinations, or all permutations, $\pi_k$, in order to find the combination with the maximum agreement. It is searching for the collection of ballots which produces the maximum minimum edge agreement.

The primal linear program is:

$$
\text{max } \alpha \\
\text{subject to } \\
\sum_{k=1}^{n!} x_k = 1 \\
\sum_{k=1}^{n!} \pi_k x_k \geq \alpha, \forall e \in E \\
x_k \geq 0, \forall k = 1, \ldots, n!
$$

The objective function, $\text{max } \alpha$, is to maximize the agreement.

The first constraint $\sum_{k=1}^{n!} x_k = 1$ requires that the proportional weighting given to each permutation sum to 1. In many cases, some permutations
are given a weighting of 0. In our example above where we constructed the
tournament with the single ballot \((ABCD)\), that permutation has \(x_k = 1\)
and all other permutations have \(x_k = 0\).

There are \(\binom{n}{2}\) of the second set of constraints, \(\sum_{k=1}^{n!} \pi_k^e x_k \geq \alpha, \forall e \in E\). These constraints force the pairwise agreement to be greater than or equal
to alpha for every permutation/ballot with a nonzero \(x_k\) weighting.

The third set of constraints, \(x_k \geq 0\), are nonnegativity constraints. Com-
bining these with the first, we see that \(x_k \in [0, 1] \quad \forall k\).

The dual is for the above primal is:

\[
\begin{align*}
\min & \quad \lambda \\
\text{subject to} & \\
& \sum_{e \in E} \mu_e = 1 \\
& \sum_{e \in E} \pi_k^e \mu_e \leq \lambda \quad \forall k = 1, \ldots, n! \\
& \mu_e \geq 0 \quad \forall e \in E
\end{align*}
\]

We propose that there is a direct connection between the dual LP and the
facet-defining inequalities of the linear ordering polytope. This connection
will be explored in the next chapter.
Chapter 5

Connecting Tournament LPs and $L^n$

This chapter connects the dual of the tournament LP introduced in the previous chapter to facets of the linear ordering polytope, and motivates a question about the strength of the connection.

Recall that the dual LP depends on $e \in E$. The set $E$ is composed of the directed edges which are present in the voting tournament. Since there are two possibilities for the directed edge between every node, there are $2^{\binom{n}{2}}$ possibilities for $E$.

Therefore, there are $2^{\binom{n}{2}}$ distinct tournament LPs corresponding to the tournaments on $n$ vertices.

**Definition.** Let us call the set of tournament LPs on $n$ vertices LP-$n$. 

70
CHAPTER 5. LPS AND $L^N$

We will be discussing the connection between LP-$n$ and $L^n$.

**Definition.** Let $(\mu_1, \mu_2, \mu_3, \ldots, \mu_{\binom{n}{2}}, \lambda)$ be a solution to LP-$n$. Then we will define an inequality in $2^{\binom{n}{2}}$:

$$\mu \cdot x \leq \lambda,$$

where $\mu \in \mathbb{R}^{2\binom{n}{2}}$ and all components of $\mu$ not in the set of variables of the LP are 0. For convenience, we can represent this inequality as a vector $(\mu, \lambda) \in \mathbb{R}^{2\binom{n}{2}+1}$.

**Example** Consider a specific instance of LP-3 where the $\mu$ variables with nonzero coefficients are $\{\mu_{12}, \mu_{13}, \mu_{23}\}$ and a solution is $(1, 0, 0, 1)$. Then this solution uniquely maps to the inequality $\mu_{12}x_{12} \leq 1$ in Euclidean 6-space.

**Lemma 1.** In the description of $L^n$ in $2\binom{n}{2}$-space, if $i, j$ are two elements in the set being permuted, then for the components in a vector in $L^n$:

$$x_{ij} = 1 - x_{ji}.$$

**Proof.** Since permutations are transitive, if $\pi(i) > \pi(j)$, then it is not possible for $\pi(j) > \pi(i)$. Thus, $x_{ij} = 1$ or $x_{ji} = 1$ but not both. The other must be 0 by the way the presence-absence components are defined in $L^n$.

So, $x_{ij} + x_{ji} = 1$. This gives both $x_{ij} = 1 - x_{ji}$ and $x_{ji} = 1 - x_{ij}$. 
This shows that every permutation vector, $x_{ij} = 1 - x_{ji}$ and $x_{ji} = 1 - x_{ij}$.

Since any other point in $L^n$ can be written as the convex combination of permutation vectors, every point in $L^n$ lies on the hyperplane defined by $x_{ij} + x_{ji} = 1$, so $x_{ij} = 1 - x_{ji}$ and $x_{ji} = 1 - x_{ij}$ is true for any element of $L^n$.

**Theorem 14.** Every valid inequality, $\mu \cdot x \leq \lambda$, for $L^n$ in $2\binom{n}{2}$-space can be written with $\mu \geq 0$, and expressed where at least one of every pair $\mu_{ij}$ and $\mu_{ji}$ is zero.

**Proof.** Assume we have a valid inequality for $L^n$, $(\mu, \lambda)$. For each case, without loss of generality, assume any nonspecified pair of $x_{ij}$ satisfy the lemma.

**Case 1:** There is a pair of $i, j$ such that $\mu_{ij} < 0$ and $\mu_{ji} \geq 0$.

We can use Lemma 1 to replace $x_{ij}$ with $1 - x_{ji}$.

Then we have an equivalent valid inequality where $x_{ij} = 0$ and $x_{ji} > 0$.

**Case 2:** There is a pair of $i, j$ such that both $\mu_{ij}, \mu_{ji} < 0$.

Assume $\mu_{ij} \leq \mu_{ji}$, which implies $|\mu_{ji}| \geq |\mu_{ij}|$. We can satisfy the lemma by replacing $x_{ij}$ with $1 - x_{ji}$. Since we replace $x_{ij}$ with $1 - x_{ji}$, we are left with a $\mu_{ij} = 0$. And since $\mu_{ij} \leq \mu_{ji}$, the coefficient $\mu_{ji}$ will be positive after multiplying.

**Case 3:** There is a pair of $i, j$ such that $\mu_{ij}, \mu_{ji} > 0$. 
CHAPTER 5. LPS AND $L^N$

Assume $\mu_{ij} \leq \mu_{ji}$. Then we can replace $x_{ij}$ with $1 - x_{ji}$, and we will be left with an equivalent inequality with $x_{ij} = 0$ and $x_{ji} \geq 0$.

Therefore we can always rewrite a valid inequality with all-nonnegative values of $\mu_{ij}$, where at least one of every pair $\mu_{ij}$ and $\mu_{ji}$ is zero.

\[ \text{Lemma 2. The maximum value for } \lambda \text{ in any instance of LP-n is 1, and the minimum is 0.} \]

\[ \text{Proof. Recall LP-n:} \]
\[ \min \quad \lambda \]

subject to
\[ \sum_{e \in E} \mu_e = 1 \]
\[ \sum_{e \in E} \pi_e^k \mu_e \leq \lambda \quad \forall k = 1, ..., n! \]
\[ \mu_e \geq 0 \quad \forall e \in E \]

Every value for \( \pi_e^k \) is either 1 or 0, and every \( \mu_e \geq 0 \). So at most,
\[ \max (\sum_{e \in E} \pi_e^k \mu_e) = \sum_{e \in E} \mu_e = \lambda. \]

Since the objective is to minimize \( \lambda \), it must be true that \( \lambda \leq 1 \).

On the other hand, since \( \pi_e^k \geq 0, \mu_e \geq 0 \),
\[ 0 = \min \sum_{e \in E} \pi_e^k \mu_e \leq \sum_{e \in E} \pi_e^k \mu_e \leq \lambda. \]

So \( 0 \leq \lambda \leq 1 \).

\[ \square \]

Let us look at a specific example of \( L^n \) and its corresponding LPs.
5.1 The Correspondence when $n = 3$

We want to show every valid inequality of $L^3$ is generated by at least one BFS from LP-3.

First, let us look at $L^3$ again. See Figure 5.1.

There are eight facets of $L^3$. In the “standard” description of $L^n$, the description of the facets depends on the choice of edge set. Note that there is one description using all $2\binom{n}{2}$ axes. Using the edge set $E = \{AB, CB, CA\}$, the permutation vectors corresponding to $E$ are:

$$P = \{(1,1,0), (1,0,0), (0,1,0), (0,1,1), (1,0,1), (0,0,1)\},$$

and the facets are defined by

\[
\begin{align*}
x_1 &\leq 1 \\
x_2 &\leq 1 \\
x_3 &\leq 1 \\
x_1 &\geq 0 \\
x_2 &\geq 0 \\
x_3 &\geq 0 \\
x_1 + x_2 + x_3 &\leq \frac{2}{3} \\
x_1 + x_2 + x_3 &\geq \frac{1}{3}
\end{align*}
\]
Figure 5.1: The linear ordering polytope where $n = 3$.

up to scalar multiplication.

These can be found by inspection. We can check by, first, showing that each of these inequalities is a valid inequality for the convex hull of all permutation vectors, and second, by showing that there are three affinely independent points in $L^3$ that lie on each inequality.

We will use LP-3 to pick out 8 BFS which map to all 8 facet-defining inequalities.

There are $2^3(3) = 8$ tournaments in the set LP-3. Each of these tournaments is isomorphic to one of the two graphs in Figure 5.2.

For the cyclic tournament, the edge set is $E = \{AB, BC, CA\}$. The
permutation vectors are

\[ P = \{(1, 1, 0), (1, 0, 0), (0, 1, 0), (0, 1, 1), (1, 0, 1), (0, 0, 1)\}. \]

So this specific LP is:

\[
\begin{align*}
\text{min} & \quad \lambda \\
\text{subject to} & \\
\mu_{AB} + \mu_{BC} + \mu_{CA} & = 1 \quad (1) \\
\mu_{AB} + \mu_{BC} & \leq \lambda \quad (2) \\
\mu_{AB} & \leq \lambda \quad (3) \\
\mu_{BC} & \leq \lambda \quad (4) \\
\mu_{BC} + \mu_{CA} & \leq \lambda \quad (5) \\
\mu_{AB} + \mu_{CA} & \leq \lambda \quad (6) \\
\mu_{CA} & \leq \lambda \quad (7) \\
\mu_e & \geq 0 \quad \forall e \in E \quad (8)
\end{align*}
\]
CHAPTER 5. LPS AND $L^N$

From this LP, we need to find basic feasible solutions. A solution will be a BFS if it is feasible (valid for every constraint) and met at equality for $\binom{n}{2} + 1 = 4$ constraints, the number of variables in any LP-3, including $\lambda$.

We know $0 \leq \mu_e \leq 1$ from constraints (1) and (8). If we set one $\mu_e = 1$ and the rest to 0, we find 3 BFS. In each situation, the minimal/optimal $\lambda$ is 1. In the form $(AB, BC, AC, \lambda)$, these BFS are:

$$(1, 0, 0, 1), (0, 1, 0, 1), (0, 0, 1, 1).$$

We also have the BFS $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{2}{3})$ since none of the constraints (2) - (7) have three nonzero $\mu$’s. This gives us four BFS.

The set $E$ for the acyclic tournament in Figure 5.2 is $E = \{AB, CB, CA\}$. The permutation vectors corresponding to $E$ are

$$P = \{(1, 0, 0), (1, 1, 0), (0, 0, 0), (0, 0, 1), (1, 1, 1), (0, 1, 1)\}.$$
So another specific LP is:

\[
\begin{align*}
\min & \quad \lambda \\
\text{subject to} & \\
\mu_{AB} + \mu_{CB} + \mu_{CA} & = 1 \quad (1) \\
\mu_{AB} & \leq \lambda \quad (2) \\
\mu_{AB} + \mu_{CB} & \leq \lambda \quad (3) \\
0 & \leq \lambda \quad (4) \\
\mu_{CA} & \leq \lambda \quad (5) \\
\mu_{AB} + \mu_{CB} + \mu_{CA} & \leq \lambda \quad (6) \\
\mu_{CB} + \mu_{CA} & \leq \lambda \quad (7) \\
\mu_e & \geq 0 \quad \forall e \in E \quad (8)
\end{align*}
\]

We have 3 BFS where one \( \mu_e = 1 \) and the other are 0. One of these is a new BFS, since we are working with the edge set with \( CB \) instead of \( BC \) as we were in equation 5.1. We will re-write the corresponding valid inequality using the ideas in Theorem 1.

These two LPs give us five facet-defining inequalities. From Equation 5.1 we have

\[
\begin{align*}
& x_{AB} \quad \leq 1 \\
& x_{BC} \quad \leq 1 \\
& x_{CA} \quad \leq 1 \\
& \frac{1}{3} x_{AB} + \frac{1}{3} x_{BC} + \frac{1}{3} x_{CA} \quad \leq \frac{2}{3}
\end{align*}
\]
From Equation 5.2 we have the additional inequality $x_{CB} \leq 1$.

By Theorem 14, we can put the BFS from Equation 5.2 in terms of the variables from Equation 5.1. This gives the valid inequality $1 - x_{BC} \leq 1$ which is the same as $x_{BC} \geq 0$.

So far we have found five of the eight facet-defining inequalities.

Using the other six LPs and Theorem 14, we can find the rest of the valid inequalities.

5.2 Facets of $L^n$

**Theorem 16.** Every facet of $L^n$ corresponds to a BFS for at least one member of LP-$n$.

**Proof.** Assume we have a facet-defining inequality for $L^n$, $f = (\mu, \lambda)$ such that $\mu \in \mathbb{R}^{2^n}$ as defined in Definition chapter 5.

Without loss of generality, using Theorems 14 and 15, assume $\mu \geq 0$ and $\sum \mu_{ij} = 1$.

Since $f$ is facet-defining, it is a valid inequality for $L^n$, and therefore any point $p \in L^n$ satisfies $\mu \cdot p \leq f$. So for any permutation vertex of $P$, written as $\pi^k$, we know $\mu \cdot \pi^k \leq \lambda$. After rewriting using Theorem 14, we have a set of $n!$ constraints $\sum_{e \in E} \pi^k_e \mu_e \leq \lambda$ corresponding to the $n!$ permutation vectors $\pi^k$ for some specific LP-$n$. 
Thus, any facet- corresponds to a feasible solution to some linear program in LP-\(n\).

Now we want to show that \(f\) gives a solution that is basic.

Since \(f\) is facet-defining, by definition, there must exist \(\text{dim}(L^n) = \binom{n}{2}\) affinely independent vertices of \(L^n\) met at equality for \(f\). This means there are \(\binom{n}{2}\) affinely independent vertices, \(\pi^k\), in \(L^n\) that satisfy \(\mu \pi^k = \lambda\).

Therefore we have \(\binom{n}{2}\) constraints of the form \(\sum_{e \in E} \pi^k_e \mu_e = \lambda\). Since the vectors were affinely independent, these constraints must all be linearly independent.

This gives us \(\binom{n}{2}\) constraints met at equality ("active") in LP-\(n\). In order for the solution to be basic, since there are \(\binom{n}{2} + 1\) variables in LP-\(n\), including \(\lambda\), we need \(\binom{n}{2} + 1\) active constraints in the solution.

In addition to those of the form \(\sum_{e \in E} \pi^k_e \mu_e = \lambda\), the convexity constraint is always active. Therefore we have \(\binom{n}{2} + 1\) active constraints. Thus the solution is basic.

Therefore, any facet-defining inequality of \(L^n\) corresponds to a BFS of the linear program. \(\square\)

We would like to know if it is true that a vector is a BFS of LP-\(n\) if and only if it corresponds to a facet-defining inequality of \(L^n\).

We have shown that the forward direction is true: every facet-defining
inequality maps to a basic feasible solution of LP-$n$. The other direction is not yet proved:

**Question 1.** Every BFS of LP-$n$ maps to a facet-defining inequality for $L^n$.

Here is what we know so far:

First, we must show that any BFS of LP-$n$ maps to a valid inequality for $L^n$.

Assume we have a basic feasible solution, $b = (\mu, \lambda)$. Then $b$ is feasible for LP-$n$ by definition of a BFS. This means every constraint in the LP is satisfied.

We know that the set of all $\pi_e^k$ in the LP are exactly the vector representations of all permutations of a given set. So the solution generates an inequality which is valid for every permutation. This means the inequality must be valid for every vertex of $L^n$.

Since any other point in $L^n$ is a convex combination of the vertices, the inequality must also be valid for any point in $L^n$. Therefore the inequality is valid for all of $L^n$.

Next we must show the BFS is facet-defining.

In order for a valid inequality for be facet-defining, it must meet $\dim(L^n) = \binom{n}{2}$ affinely independent points at equality.
By definition of a BFS, there must be at least as many constraints satisfied at equality ("active") as there are variables. Including $\lambda$, LP-$n$ contains $\binom{n}{2} + 1$ variables. Thus, there must be at least $\binom{n}{2}$ + 1 active constraints.

We know one of those constraints is the convexity constraint, $\sum_{e \in E} \mu_e = 1$.

This is where the proof falls apart. We need $\binom{n}{2}$ active constraints of the form $\sum_{e \in E} \pi^k_e \mu_e$ so that they correspond to $\binom{n}{2}$ permutation vectors met at equality by the inequality mapped to by $b$.

This is an issue because it may be possible that a BFS includes fewer than $\binom{n}{2}$ active constraints of the form $\sum_{e \in E} \pi^k_e \mu_e$. They may include some active constraints of the form $\mu_e \geq 0$, the nonnegativity constraints.

Our conjecture is that any BFS which includes active constraints of the form $\mu_e \geq 0$ is a degenerate BFS, meaning it has more than $\binom{n}{2}$ active constraints, and still has $\binom{n}{2}$ active constraints of the form $\sum_{e \in E} \pi^k_e \mu_e$. But this is yet to be proved.

If we knew the nonnegativity constraints in LP-$n$ were redundant, then we could easily show that every BFS generates a facet of $L^n$. That is because if the feasible region of LP-$n$ is identical both with and without the nonnegativity constraints, all vertices would have $\binom{n}{2}$ active permutation constraints.

This is not the case.

**Theorem 17.** Nonnegativity constraints are necessary in the description of
the feasible region of LP-n.

Proof. By counterexample.

Consider the linear program in $LP-3$ given in Equation 5.1, but without nonnegativity constraints:

\[
\begin{align*}
\min & \quad \lambda \\
\text{subject to} & \\
\mu_{AB} + \mu_{BC} + \mu_{CA} &= 1 \quad (1) \\
\mu_{AB} + \mu_{BC} &\leq \lambda \quad (2) \\
\mu_{AB} &\leq \lambda \quad (3) \\
\mu_{BC} &\leq \lambda \quad (4) \\
\mu_{BC} + \mu_{CA} &\leq \lambda \quad (5) \\
\mu_{AB} + \mu_{CA} &\leq \lambda \quad (6) \\
\mu_{CA} &\leq \lambda \quad (7)
\end{align*}
\]

Let $\mu_{CA} < 0$.

Due to the convexity constraint, $\mu_{AB} + \mu_{BC} + \mu_{CA} = 1$ implies

$$\mu_{AB} + \mu_{BC} = 1 - \mu_{CA} > 1$$

Then is we replace $\mu_{AB} + \mu_{BC}$ in constraint (2) with this inequality, we see:
Thus, the minimum value of $\lambda$, or the objective value, must be greater than 1.

Since the maximal optimal value of $\lambda$ is 1 by Lemma 2, we have constructed a solution outside the feasible region for $LP - n$. Therefore we have reached a contradiction, and see the nonnegativity constraints are necessary in the description of $LP - n$.

The above argument works for any LP which contains at least one constraint where $\pi^k_e = 0$ for one $e$ and $\pi^k_e = 1$ for all others.

Even if it is not true that every BFS of $LP - n$ maps to a facet-defining inequality of $L^n$, it may be true that every optimal BFS maps to a facet-defining inequality of $L^n$. This would still provide insight into the facets of $L^n$.

There is still work to be done, but if we can prove that even some of the BFS of $LP - n$ give a facet-defining inequality of $L^n$, then we may be able to write down a new class of facets of $L^n$ as defined by these tournament LPs. Ultimately, this could be used to improve the computational runtime of algorithms utilizing the linear ordering polytope and the linear ordering problem to find solutions.
Chapter 6

Conclusion

6.1 Future Work

The primary focus of future work is to explore whether every BFS of LP-$n$ corresponds to a valid facet-defining inequality for $L^n$.

In an ideal situation, we hope to fully understand the correspondence between the BFS of LP-$n$ and the facet-defining inequalities of $L^n$. If that is possible, it may lead to a better understanding of the classes of facets of $L^n$, and ultimately an improvement in the runtime of discrete optimization solvers related to orderings.

Additional future work may focus on another polytope, the permutohedron, $P^n$. The linear ordering polytope and the permutohedron are both polytopes constructed as the convex hull of permutation vectors. The differ-
ence begins in the way the permutation vectors are constructed.

For $L^n$, a permutation vector consists of \{x_{12}, x_{13}, x_{14}, ..., x_{1n}, x_{23}, x_{24}, ..., x_{(n-1)n}\}

where

$$x_{ji} = \begin{cases} 1 & \text{if } i > j \\ 0 & \text{else.} \end{cases}$$

For $P^n$, the $k$th component of the permutation vector is exactly the $k$th element of the permutation.[19]

Some basic comparisons of the two polytopes:

<table>
<thead>
<tr>
<th></th>
<th>$L^n$</th>
<th>$P^n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>the permutation (123) maps to</td>
<td>(1,1,1)</td>
<td>(1,2,3)</td>
</tr>
<tr>
<td>the permutation (1234) maps to</td>
<td>(1,1,1,1,1,1)</td>
<td>(1,2,3,4)</td>
</tr>
<tr>
<td>dimension of permutation vectors =</td>
<td>$\frac{n!}{2}$</td>
<td>$n$</td>
</tr>
<tr>
<td>$\text{dim}(L^n)$ or $\text{dim}(P^n)$</td>
<td>$\frac{n!}{2}$</td>
<td>$n - 1$</td>
</tr>
</tbody>
</table>

Notably, the dimension of $P^n = n - 1$ since the polytope lies entirely on the hyperplane $\sum_{k=1}^{n} k = \sum_{k=1}^{n} x_k$, $k \in \mathbb{R}$, $x = (x_1, x_2, ..., x_n) \in P^n$.

A projection was defined from $L^n$ to $P^n$, then from $P^n$ to $L^{n-1}$ in Katthän’s *The Linear Ordering Polytope via Representations*.[7] It may be fruitful to also look for connections between the facets and geometry of $L^n$ and $P^n$ to see if that could also improve our understanding of the facets of $L^n$. 

CHAPTER 6. CONCLUSION

6.2 Summary

The linear ordering polytope is a convex polytope constructed from the convex hull of a set of permutation vectors. The polytope is important in discrete optimization problems such as the linear ordering problem and the travelling salesman problem, yet the facet structure of the linear ordering polytope is not fully classified.

By developing an understand of the basic feasible solution to a specific voting optimization problem related to voting tournaments, we have shown a direct mapping from the facet-defining inequalities of the linear ordering polytope to the basic feasible solutions to the optimization problem.
If, in the future, we can show that there is also a connection from every basic feasible solution to the facet-defining inequalities, we may be able to understand another class of facets of the linear ordering polytope.

Another class of facets would be instrumental in decreasing the runtime of combinatorial optimization problems which rely on $L^n$, as they may be able to be used to generate cutting planes for optimization problems whose feasible region is $L^n$. Therefore, the big question is, can connections between the dual of the voting LP be used to generate cutting planes for these specific optimization problems?

This thesis has begun to explore the connection, but there is further work to be done before the above question can be answered.
Bibliography


