Solving IP with LP

Drew Morris

I. IP-LP Correspondence

In this section, we will show how one can use LP to solve an integer programming problem. Consider a linear programming problem of the following form.

$$\begin{array}{llcl} \max_{\mathbf{x}\in\mathbb{R}^n} & \mathbf{c}^T\mathbf{x} & & \\ \text{subject to} & A\mathbf{x} & \leq & \mathbf{b} \\ & \mathbf{x} & \geq & \mathbf{0} \end{array}$$

where $A \in M_{m\times n}(\mathbb{R})$, $\mathbf{b} \in \mathbb{R}^m$, and $\mathbf{c} \in \mathbb{R}^n$, for some $m, n \in \mathbb{N}_0$. The corresponding integer programming problems can be written in the following form.

$$\begin{array}{llcl} \text{maximize} & \mathbf{c}^T\mathbf{x} & & \\ \text{subject to} & A\mathbf{x} & \leq & \mathbf{b} \\ & \mathbf{x} & \geq & \mathbf{0} \\ & \mathbf{x} & \in & \mathbb{Z}^n \end{array}$$

Notice that the only difference between these problems are the domain of their decision variable(s) ($\mathbb{R}^n$,$\mathbb{Z}^n$ respectively). We will use this fact to show how one could use LP to help solve an integer programming problem.

For the remainder of this paper we will take $X^* \subseteq X \subseteq \mathbb{R}^n$ to be the optimal and feasible solution spaces for a given linear programming problem, respectively. Similarly, we will take $Z^* \subseteq Z \subseteq \mathbb{Z}^n$ to be the optimal and feasible solution spaces for the IP correspondent.

A. IP-LP Minimaxity

We will our integer programming problem by solving a problem equivalent to it that is built around minimaxity. Recall our given linear programming problem and observe the equivalent (although trivial) equivalent problem using minimaxity.

$$\begin{array}{llcl} \text{minimize} & ||\mathbf{x}' - \max_{\mathbf{x}\in X} \mathbf{c}^T\mathbf{x}||_2 & & \\ \text{subject to} & A\mathbf{x} & \leq & \mathbf{b} \\ & \mathbf{x}, \mathbf{x}' & \geq & \mathbf{0} \end{array}$$

i.e.

$$\min_{\mathbf{x},\mathbf{x}^*\in X, X^*} ||\mathbf{x} - \mathbf{x}^*||_2$$

This problem (on its own) is trivial because it requires one to have already solved the linear programming problem it is derived from to solve. Semantically, it says, ”To solve a linear programming problem, first solve your linear programming problem and then take the set of solutions in X closest to the optima you found.” However, this trivial problem's IP coresepondent is not trivial. Observe.

$$\begin{array}{llcl} \text{minimize} & ||\mathbf{x}' - \max_{\mathbf{x}\in X} \mathbf{c}^T\mathbf{x}||_2 & & \\ \text{subject to} & A\mathbf{x} & \leq & \mathbf{b} \\ & \mathbf{x}, \mathbf{x}' & \geq & \mathbf{0} \\ & \mathbf{x}' & \in & \mathbb{Z}^n \end{array}$$

i.e.

$$\min_{\mathbf{z},\mathbf{x}^*\in Z, X^*} ||\mathbf{z} - \mathbf{x}^*||$$

Semantically, this says, ”To solve an integer programming problem, first solve its LP correspondent and then take the set of solutions in Z closest to the optima you found.” With this we have found the following algorithm for solving an integer programming problem using LP.

1) Solve the LP correspondent.
2) Exhaustively compute $||\mathbf{z} - \mathbf{x}^*||_2$ for every $\mathbf{z} \in Z$ and $\mathbf{x}^* \in X^*$.
3) Take the set of $\mathbf{z}^*$(s) yielding minimal results.

This is a very inefficient algorithm, but we can improve it by bounding the feasible solution space of our integer programming problem, Z. However, we must first prove some important properties of IP and LP and their feasible and optimal solution spaces.

II. IP-LP Solution Space Properties

In this section, we will prove some important properties of X, X^*, Z, Z^* to the end of using those properties to improve our algorithm.

A. LP Convexity and Polytopicity

Let $X^* \subseteq X$ be the set of optimal solutions to a given linear programming problem.

Proposition II.1. *X^* is a convex polytope.*

Proof. Suppose $\hat{\mathbf{x}}_1 \in X^*$. If $\hat{\mathbf{x}}_1$ is unique then $X^* = \{\hat{\mathbf{x}}_1\}$ and we are done. Thus, suppose $\hat{\mathbf{x}}_2 \in X^*$ such that $\hat{\mathbf{x}}_2 \neq \hat{\mathbf{x}}_1$. Then $t\hat{\mathbf{x}}_1 + (1-t)\hat{\mathbf{x}}_2 \in X^*$for every $t \in [0,1]$. As such we take $\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2$ such that they maximize the euclidean metric across the region. If this region spans X^* then we are done, otherwisewe can repeat this process countabily many times until we have a convexly independent set of points, $\hat{X} = \{\hat{\mathbf{x}}_1, ..., \hat{\mathbf{x}}_k\}$, such that every point, $\mathbf{x} \in X^*$, can be expressed as a convex combination of $\hat{X}$ i.e. for every $\mathbf{x} \in X^*$, there exists $\mathbf{t} \in \mathbb{R}^k$, such that $\mathbf{t} \geq \mathbf{0}, ||\mathbf{t}||_1 = 1$, and

$$\begin{bmatrix} \hat{\mathbf{x}}_1 & ... & \hat{\mathbf{x}}_k \end{bmatrix} \mathbf{t} = \mathbf{x}$$

which describes a convex polytope of k vertices, and subsequently, at most $k-1$ dimension.

□

Remark II.2. For a linear programming problem of n decision variables and m constraints, X^* will be a polytope of at most $m+n$ vertices and at most n dimension. We will prove this remark further on in this paper.

As the above is a computable process, the above proof is sufficient to prove that said process can be used for any linear programming problem of up to countably infinitely many constraints and or decision variables. However, by LP being based in linear algebra, which does allow for the existence of vector spaces of uncountably infinite dimension, the above proof is insufficient to prove X^* is a convex polytope for linear programming problems of uncountably many contraints and or decision variables.

As such, for the remainder of this paper it is assumed that we are working with linear programming problems of only countably many constraints and or decision variables.

Corollary II.3. *X is a convex polytope.*

Proof. The proof X being a convex polytope follows the same logic as the proof for X^*.

□

Remark II.4. If X is bounded non-redundantly (meaning that the removal of any constraint would alter the feasible region, X) then X has $m+n$ vertices (assuming that we are including the n non-negativity constraints in our set of constraints). This follows from X being a convex polytope. More interestingly, if X has only $m'+n$ binding (non-redudant) constraints, for some $0 \leq m' < m$, then X is a convex polytope of $m'+n$ vertices. This is a consequence of being able to remove redundant constraints without altering the feasible region.

Remark II.5. If X is decided linearly independently (meaning that there is no subset of constraints that reduces the dimension of the feasible region, X, to less than the dimension of the decision variable by forcing any "feasible" region of n decision variables to be linearly dependent) then X is a polytope of n dimension. More interestingly, if X has only n' linearly independent decision variables, for some $0 \leq n' < n$, then X is a convex polytope of n' dimension. This is a consequence of being able to project our feasible region, its constraints, and our objective function down to n' decision variables, which is n'-dimensional, without changing the ordering of the feasible region by nature of lossless (or non-colliding) projection being an isomorphism.

Given these two remarks, for the remainder of this paper, without loss of generality, we assume that we are working with linear programming problems with non-redudant constraints and linearly independent decision variables.

B. IP Integral Convexity and Polytopicity

Let $Z^* \subseteq Z$ be the set of optimal solutions to a given integer programming problem.

Proposition II.6. *Z^* is an integrally convex polytope.*

Proof. By applying the same logic from the proof Proposition II.1., we find a convex polytope, S, with integer vertices, $\hat{Z}$, such that $Z^* \subseteq S$. The non-equivalence S and Z^* is due to S (possibly) containing vectors that are not in $\mathbb{Z}^n$ by nature of containing all convex combinations of $\hat{Z}$. We can resolve this by restricting S to $\mathbb{Z}^n$. Showing,

$$S\mid_{\mathbb{Z}^n} = S \cap \mathbb{Z}^n = Z^*$$

which is the set of all integrally convex combinations of $\hat{Z}$ which is, by definition, integrally convex.

□

C. IP-LP Non-Intersectionality

Let X^*, Z^* be the set of optimal solutions for a pair of correspondent linear and integer programming problems.

Proposition II.7. *X^* contains an integral optima if and only if X^* contains all integral optima.*

Proof. We wish to prove

$$\exists \mathbf{z}^* \in Z^*, \mathbf{z}^* \in X^* \iff \forall \mathbf{z}^* \in Z^*, \mathbf{z}^* \in X^*$$

i.e.

$$Z^* - X^* \neq \emptyset \iff X^* \cap \mathbb{Z}^n = \emptyset$$

We proceed by proving each part of this implication separately.

($\Longleftarrow$) Suppose $X^* \cap \mathbb{Z}^n = \emptyset$. Then, for every $\mathbf{z}^* \in Z^*$, $\mathbf{z}^* \notin X^*$ since $\mathbf{z}^* \in \mathbb{Z}^n$. Thus $Z^* - X^* = \emptyset$.

($\Longrightarrow$) Working contrapositively, suppose $X^* \cap Z^* \neq \emptyset$. Then there must exist some $\mathbf{z}^* \in Z^*$ such that $\mathbf{z}^* \in X^*$. Therefore, $\mathbf{z}^*$ is a convex combination of X^*. Now, by way of contradiction, suppose there exists $\mathbf{z}' \in Z^*$ such that $\mathbf{z}' \notin X^*$. If $\mathbf{z}^* = \mathbf{z}'$ then $\mathbf{z}^* \notin X^*$, which is a contradicion. As such, further suppose $\mathbf{z}' \neq \mathbf{z}^*$. Then $\mathbf{z}'$ is not an optima for the given linear programming problem; however, $\mathbf{z}'$ is an optima for the given integer programming problem. Therefore, either $\mathbf{z}'$ is an optima for the given linear programming problem or $\mathbf{z}^*$ is not an optima for the given linear programming problem: both of which are contradictions. Thus $Z^* - X^* = \emptyset$.

□

D. LP Vertex Cardinality

Assume we are a given a feasible, bounded, linear programming problem of m decision variables and n constraints where $m, n \in \mathbb{N}$, i.e. $A \in M_{m\times n}(\mathbb{R})$.

Proposition II.8. *X^* is a convex polytope of at least 1 vertex and at most $m+n$ vertices i.e.*

$$1 \leq |\hat{X}| \leq m+n$$

Proof. Assume the above premise. Then the given linear programming problem has at least 1 solution. Therefore $|\hat{X}| \geq 1$. Furthermore, if there exists an optimal solution to the given linear programming problem, then there must exist an optimal solution on the boundary of the feasible region, X. This boundary is defined as the intersection of $m+n$ half spaces, namely the m non-negativity constraints and the n bounding constraints. As such, the boundary of X is a convex polygon of at most $m+n$ vertices. Furthermore, if there exists an optimal solution to the given linear programming problem, then there must exist at least 1 optimal solution that is also a vertex of the boundary of X. Let $\hat{V}$ be the set of vertices on the boundary of X that are also optimal solutions. Then $|\hat{V}| \leq m+n$ and, by the convexity of X^*, X^* is equal to the convex hull of $\hat{V}$. Therefore, $\hat{X} = \hat{V}$ and, consequently, $|\hat{X}| \leq m+n$.

□

E. LP Dimensionality

Once again, assume we are a given a feasible, bounded, linear programming problem of m decision variables and n constraints where $m, n \in \mathbb{N}$, i.e. $A \in M_{m\times n}(\mathbb{R})$.

Proposition II.9. *X^* is a convex polytope of at least 0 dimension and at most n dimension.*

Proof. Assume the above premise. Since X^* is a convex polytope with 1 to $m+n$ (inclusive) vertices, $\hat{X}$, X^* must have dimension of at least $1-1=0$ and at most $m+n-1$. Furthermore, $X^* \subseteq X \subseteq \mathbb{R}^n$ so X^* must have a dimension of at most that of $\mathbb{R}^n$, which is n. Thus X^* is a polytope of at least 0 dimension and at most n dimension.

□

Using these properies, we can construct a more efficient, 2 part, algorithm.

III. Maximal Inclusion Algorithm

In this section, we develop an algorithm that will find an IP-optimum, $\mathbf{z}^*$, in the set of LP-optima, X^*, given one exists. In order to develop this algorithm, we must first recall the Intermediate Value Theorem (IVT for short).

Theorem III.1 (Intermediate Value Theorem)**.** *Let $I = [a, b] \subseteq \mathbb{R}$ be a closed, sub-interval of the real numbers and let $f : I \to \mathbb{R}$ be a continuous function across that interval. Then, for every $y \in f(I) = [\min(f(a), f(b)), \max(f(a), f(b))]$, there must exist some $x \in I$, such that $f(x) = y$.*

Suppose we have solved our linear programming problem and have found X^* and $\hat{X}$ respectively. Our algorithm to find $\mathbf{z}^*$ is built around "slicing off" the outermost edges of our polytope, X^*, in a cubic fashion. We must perform one set of "cuts" for each dimension of our decision variable.

In more formal terms, we wish to find the largest interval, $[a_i, b_i]$, such that $a_i, b_i \in \mathbb{Z}, a_i \leq b_i$ and the half spaces $v_i = a_i, v_i = b_i$ both intersect X^*. For each individual set of "cuts", this process yields a new $X^*, \hat{X}$ which we will call $X', \hat{X}'$ respectively. Furthermore, after a "cut" along the v_i-axis, X' is guaranteed to have integral edges along the v_i-axis at exactly $v_i = a_i$ and $v_i = b_i$. Most importantly, X' is guaranteed to remain convex polytope due to the convexity and polytopicity of X' and the definition of this process as applying additional half space intersections.

Before defining an algorithm about this process, we must more rigorously define what we "cutting" means. In actuality, how one might perform this "cutting" operation is by doing the following. First, project X^* down onto the v_i-axis this "removes" any data external to our current integral interval conclusion from consideration. This is written as follows.

$$S_i = \text{proj}_{E_i}(S)$$

for some convex polytope, S, (which will use for both X^* and X') where

$$E_i = \text{span}(\{\mathbf{e}_i\})$$

which can also be represented as the matrix in $M_n(\mathbb{R})$ of all 0s except for at index (i, i) where it is 1. As such, we can now solve

$$[a_i, b_i] = \text{argmax}_{[a,b] \subseteq S_i : a, b \in \mathbb{Z},} |b - a|$$

to find the maximal integral inclusion of S along the v_i-axis. In other words, we have found the subset of S of the smallest measure that contains the largest number of integers. This can be done by taking

$$[a_i, b_i] = [\lceil c_i \rceil, \lfloor d_i \rfloor]$$

where

$$[c_i, d_i] = \text{argmax}_{[a,b] \subseteq S_i} |b - a|$$

We can now project this interval up into our original space such that we can perform an intersection by finding the half spaces corresponding to the bounds of this interval. We call these half spaces $\mathcal{A}_i$ and $\mathcal{B}_i$ and we define them as follows.

$$\mathcal{A}_i = \{\mathbf{v} \in \mathbb{R}^n : v_i \geq a_i\}$$

and

$$\mathcal{B}_i = \{\mathbf{v} \in \mathbb{R}^n : v_i \leq b_i\}$$

or equivalently, using matrix notation,

$$\mathcal{A}_i = \{E_i \mathbf{v} \geq a_i \mathbf{e}_i : \mathbf{v} \in \mathbb{R}^n\}$$

and

$$\mathcal{B}_i = \{E_i \mathbf{v} \leq b_i \mathbf{e}_i : \mathbf{v} \in \mathbb{R}^n\}$$

Now that we have half spaces of the proper dimension, we can perform our "slice" by performing a set intersection. Once again, this is written as follows.

$$S' = S \cap \mathcal{A}_i \cap \mathcal{B}_i$$

Also, it is reasonable to assume, given A_i, B_i, and $\hat{S}$, that $\hat{S}'$ is not difficult to find and, therefore, we take the function,

$$V : \mathbb{R}^n_{\aleph_0} \times \mathcal{H}[\mathbb{R}^n] \to \mathbb{R}^n_{\aleph_0}$$

such that

$$V(V(\hat{S}, \mathcal{A}_i), \mathcal{B}_i) \mapsto \hat{S}' \mapsfrom V(V(\hat{S}, \mathcal{B}_i), \mathcal{A}_i)$$

where $\mathbb{R}^n_{\aleph_0}$ is the set of finite subsets of $\mathbb{R}^n$ and $\mathcal{H}[\mathbb{R}^n]$ is the set of half spaces of $\mathbb{R}^n$, without further explanation. Without loss of generality, we will use the following truncation in the construction of our algorithm.

$$V(\hat{S}, \mathcal{A}_i, \mathcal{B}_i) = V(V(\hat{S}, \mathcal{A}_i), \mathcal{B}_i)$$

Similarly, we take the function

$$\mathcal{H} : M_n(\mathbb{R}) \times \mathbb{R}^n \times \mathbb{1} \to \mathcal{H}[\mathbb{R}^n]$$

to be defined as

$$\mathcal{H}(A, \mathbf{b}, t) = \begin{cases} \text{t = 0} & \{A\mathbf{v} \geq \mathbf{b} : \mathbf{v} \in \mathbb{R}^n\} \\ \text{t = 1} & \{A\mathbf{v} \leq \mathbf{b} : \mathbf{v} \in \mathbb{R}^n\} \end{cases}$$

Notice that, as long as the solution to the maximal integral inclusion, $[a_i, b_i]$, exists, $[a_i, b_i]$ is guaranteed to contain $b_i - a_i + 1$ integers by the Intermediate Value Theorem. Thus, it is only when a solution to the maximal integral inclusion fails to exist that we can determine, with certainty, that S_i does not contain an integer solution. This is useful because, when applied to X_i^*, such an interval failing to exist shows, for every $\mathbf{x}^* \in X^*$,

$$x_i^* \notin \mathbb{Z}$$

Which, consequently, shows that X^* contains no integer solutions. contrapositively, this means if we can find maximal integral inclusions of X^* for every v_i-axis in $\mathbb{R}^n$, then we can determine that X^* must contain at least 1 integer solution and, therefore, contains all IP-correspondent solutions.

By performing this process iteratively across every v_i-axis in $\mathbb{R}^n$, we construct the following algorithm to find some $\mathbf{z}^* \in X^*$ given one exists.

$X' \leftarrow X^*$

$\hat{X}' \leftarrow \hat{X}$

`for(`$i \leftarrow 1; i \leq n; i \leftarrow i + 1$`){`

  $X_i' \leftarrow \text{proj}_{E_i}(X')$

  $c_i, d_i \leftarrow \text{argmax}_{[a,b] \subseteq X_i'} |b - a|$

  $a_i, b_i \leftarrow \lceil c_i \rceil, \lfloor d_i \rfloor$

  $\mathcal{A}_i, \mathcal{B}_i = \mathcal{H}(I, a_i\mathbf{e}_i, 0), \mathcal{H}(I, b_i\mathbf{e}_i, 1)$

  $X' \leftarrow X' \cap \mathcal{A}_i \cap \mathcal{B}_i$

  $\hat{X}' \leftarrow V(\hat{X}, \mathcal{A}_i, \mathcal{B}_i)$

`}`

`for (`$\mathbf{v} \in \hat{X}'$`){`

  `if(`$\mathbf{v} \in \mathbb{Z}^n$`){`

    `return` $\mathbf{v}$

  `}`

`}`

`return NULL`

This algorithm works by first setting $X', \hat{X}'$ to $X^*, \hat{X}$ respectively, then performing our aformentioned process on $X', \hat{X}'$, and then updating $X', \hat{X}'$ to the results of that, for every v_i-axis in $\mathbb{R}^n$. Assuming $Z^* \subseteq X^*$, this algorithm will produce X', such that $Z^* \subseteq X' \subseteq X^*$ and at least 1 vertex, $\hat{\mathbf{x}}' \in \hat{X}'$, such that $\hat{\mathbf{x}}' \in \hat{Z}$.

IV. An Alternate View of LP

Before we construct the second part of our algorithm, we will first analyze a different way to look at and solve linear programming problems. Suppose we have been given a bounded, feasible, linear programming problem of the following form.

$$\begin{array}{lrcl} \text{maximize} & \mathbf{c}^T\mathbf{x} & & \\ \text{subject to} & A\mathbf{x} & \leq & \mathbf{b} \\ & \mathbf{x} & \geq & \mathbf{0} \end{array}$$

A. Affine Linear Spaces

Notice

$$\{\mathbf{x} \in \mathbb{R}^n : \mathbf{c}^T\mathbf{x} = 0\}$$

is a generalized form of a hyperplane of dimension $n - 1$ in $\mathbb{R}^n$ (known as vector form) and similarly

$$\{\mathbf{x} \in \mathbb{R}^n : \mathbf{c}^T(\mathbf{x} - \mathbf{x}_0) = 0\}$$

is a generalized form of an affine hyperplane of dimension $n - 1$ in $\mathbb{R}^n$ (again, known as vector form).

Furthermore,

$$\{t\mathbf{c} : t \in \mathbb{R}\}$$

is a generalized form of the orthogonally complementary, linear subspace to both of the above (once again, known as vector form).

B. Implicit Form

Now recall that we are trying to find the maximal solution for

$$\mathbf{c}^T\mathbf{x}$$

such that

$$A\mathbf{x} \leq \mathbf{b}$$

and

$$\mathbf{x} \geq \mathbf{0}$$

So far, we have approached solving this problem through explicit approaches: choosing $\mathbf{x}$ values that yield higher and higher results until we have hit a maximum. Now, instead, consider the following. The above problem is equivalent to solving

$$\max_{\zeta \in \mathbb{R}} \zeta$$

such that

$$\{\mathbf{x} \in \mathbb{R}^n : \mathbf{c}^T\mathbf{x} = \zeta, A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}\} \neq \emptyset$$

This is the implicit form of our linear programming problem.

C. Implicit Forms and Affine Linearity

Now notice that for every $\zeta \in \mathbb{R}$, there exists a $\mathbf{x}_0 \in \mathbb{R}^n$ such that

$$\{\mathbf{x} \in \mathbb{R}^n : \mathbf{c}^T\mathbf{x} = \zeta\} = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{c}^T(\mathbf{x} - \mathbf{x}_0) = 0\}$$

and vice versa. This is because

$$\{\mathbf{x} \in \mathbb{R} : \mathbf{c}^T\mathbf{x} = \zeta\}$$

is also a generalized form of an affine hyperplane of dimension $n - 1$ in $\mathbb{R}^n$ (known as standard form). Importantly, this shows for any choice of $\mathbf{x}_0 \in \mathbb{R}^n$, every $\mathbf{x}$ that lies on the corresponding affine hyperplane has the objective value. Because of this, we can adopt the following intuition for solving the implicit form of our linear programming problem.

V. THE DATING GAME

A. The Game

Suppose we have fixed some $\mathbf{b}, \mathbf{c}, \mathbf{x}_0 \in \mathbb{R}^n$ and some $A \in M_n(\mathbb{R})$. Now we are ready to set up our game. Suppose we have 2 players, Estella and Pip. We give each of them a set of points in $\mathbb{R}^n$. Specfically, we give Pip

$$S = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{c}^T(\mathbf{x} - \mathbf{x}_0) = 0\}$$

and Estella,

$$T = \{\mathbf{x} \in \mathbb{R}^n : A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}, \mathbf{c}^T(\mathbf{x} - \mathbf{x}_0) \geq 0\}$$

The game plays out as follows. Pip and Estella choose points, $\mathbf{s} \in S$ and $\mathbf{t} \in T$ respectively. Estella is then wins points equal to

$$||\mathbf{t} - \mathbf{s}||_2$$

and Pip loses the same number of points. This game is blatantly unfair, but it is also a 0-sum game. Therefore, there exists an optimal strategy for both Pip and Estella. These optimal strategies will help us solve our linear programming problem.

B. Connection to LP

Notice, that Pip's plays are exactly the choices of decision variables for our linear programming problem that yield a specific objective function value. Choosing different values for ζ is equivalent to sliding the space of Pip's plays along the line orthogonal to said space intersecting the origin. In our linear programming problem, we wish to maximize this value, whilst remaining within the bounded region. This is equivalent to minimizing the maximal distance between Pip's play space and Estella's play space. In other words, solving our linear programming problem is equivalent to perpendicularly shifting Pip's play space such that our game becomes fair. We will achieve this my taking smaller and smaller subsets of our feasible region that are guaranteed to contain the optimal region.

C. Maximal Orthogonal Translation Bounding

For any given suboptimal value of ζ, the optimal value, ζ^*, with optimal solution space, X^*, is guaranteed to lie above our current solution space, S, corresponding to the suboptimal object value, ζ. As such, we can update our bounded region by intersecting it with the boundary, S, to restrict our solution space while preserving its containment of the optimal solution space, which will result in a more fair game. We can perform this boundary intersection procedure by translating the objective function along its orthogonal complement, until reaching maximal inclusion in the bounded region (as previously discussed), and taking the intersection of our current boundary with this translation to be our updated boundary. Then translating the orthogonal complement along the objective function, until reaching its maximal inclusion in the updated bounded region. We repeat these two translational processes until one translation fails to change our boundary, indicating that we have encountered a steady state. At this point, the region of all maximal inclusions of the orthogonal complement will be the prismatic polytope whose lower bounding face, which we will call X', is our current objective function solution space and the upper bounding face is the optimal solution space, X^*. As such, performing one final translation of our objective function along its orthogonal complement (but this time minimizing the distance between the objective function and the boundary of the feasible region) will yield the optimal solution space.

Conjecture V.1. *The above procedure is a valid and consistent method of solving linear programming problems.*

Given that the above conjecture holds, we can now create an algorithm for finding an IP-optimum, $\mathbf{z}^*$, when X^* contains no integer solutions.

VI. Minimal Exclusion Algorithm

In this section, we develop an algorithm that will find an IP-optimum, $\mathbf{z}^*$, by minimizing the distance from some $\mathbf{z} \in Z$ to the set of LP-optima, X^*, given X^* contains no integer solutions. The motivation of this algorithm is to use the X', X^* found by the preceding algorithm to bound the IP-optima space, Z^*.

A. Minimal Vertex Balls

Suppose we have $X^*, \hat{X}^*$ from the the optimal solution to our linear programming problem. The minimal n-sphere whose boundary intersects some $\mathbf{z} \in \mathbb{Z}^n$ centered at some $\hat{\mathbf{x}}^* \in \hat{X}^*$ has at 2^{n-1} points on its boundary in $\mathbb{Z}^n$ that also lie within the feasible region. We will call this $n-$sphere, $B(\hat{\mathbf{x}}^*, r^*)$, where r^* is the measure of this n-sphere's radius. By definition, there are no points in the interior of $B(\hat{\mathbf{x}}^*, r^*)$ that are also in $\mathbb{Z}^n$.

Conjecture VI.1. *For every $\hat{\mathbf{x}}^* \in \hat{X}^*$, the integer point(s), $\bar{\mathbf{z}}$, that lie on the boundary of $B(\hat{\mathbf{x}}^*, r^*)$ occur at the same points relative to $\hat{\mathbf{x}}^*$ regardless of the value of $\hat{\mathbf{x}}^*$.*

Given that the above conjecture holds, each polytope, $\bar{Z}$, defined by the vertices of $\bar{\mathbf{z}}$ relative to each $\hat{\mathbf{x}}^* \in \hat{X}^*$ has the same objective value across the entire polytope. This is because each polytope, $\bar{Z}$, is orthogonal to the objective function fixed to a given objective value.

B. Minimally Orthogonal Polytopes

Given the validity of the previous section, we know the polytope, $\bar{Z}$, that is closest to X^* is the one defined by the vector difference, $\bar{\mathbf{z}} - \hat{\mathbf{x}}^*$, that is the least orthogonal to (or rather closest to its projection onto) X^*. We only need to check the difference of $\bar{\mathbf{z}} - \hat{\mathbf{x}}^*$ for one $\hat{\mathbf{x}}^* \in \hat{X}^*$ for each $\bar{\mathbf{z}} \in \bar{Z}$ because of the aforementioned equivalence of $\bar{\mathbf{z}}$ relative to each $\hat{\mathbf{x}}^*$. Thus we only have to check the orthogonality of $\bar{\mathbf{z}} - \hat{\mathbf{x}}^*$ for the 2^{n-1} values of $\bar{\mathbf{z}} \in \bar{Z}$ and for a single value of $\hat{\mathbf{x}}^* \in \hat{X}^*$. Then once we have the minimizing value of $\bar{\mathbf{z}}$, all we must do is find one of the $\hat{\mathbf{x}}^* \in \hat{X}^*$ such that the point $\bar{\mathbf{z}}$ relative to $\hat{\mathbf{x}}^*$ lies with in the feasible region. Thus, we have to check, at worst, $2^{n-1} + n$ values. However, I suspect that this can be improved.

C. Prismatic Polytopic Bounding

Unlike previous sections, this section is purely conjecture from playing around with the previous principles, processes and properties.

Conjecture VI.2. *Of the 2^{n-1} aforementioned polytopes, $\bar{Z}$, between 1 and n of them intersect the the previously discussed prismatic polytope defined by the faces X' and X^*.*

If the above conjecture holds, we only need to check the orthogonality of $\bar{\mathbf{z}} - \hat{\mathbf{x}}^*$ for at most n values of $\bar{\mathbf{z}} \in \bar{Z}$ for one $\hat{\mathbf{x}}^* \in \hat{X}^*$ instead of the otherwise 2^{n-1} values of $\bar{\mathbf{z}}$ which lowers the number of values we need to check to, at worst, $2n$ values.

VII. The Final Algorithm

Given that all of the previously discussed mathematics holds, our final algorithm for solving integer programming problems using LP goes as follows. First, solve the LP-correspondent, yielding X^* and $\hat{X}^*$. Perform the algorithm outlined in section III on $X^*, \hat{X}^*$. If this yields a solution, we are done, otherwise, perform the algorithm outlined in section VI on $X^*, \hat{X}^*$, which will yield a solution because the previous step did not.

A. Temporal Complexity

The motivation for the above processes is that, once again given that all of the unproven conjectures hold, every outlined algorithm is a deterministic, polynomial time, algorithm. As such, our final algorithm is the composition of deterministic, polynomial time, algorithms and is therefore a deterministic polynomial time algorithm. This would mean that IP is polynomial time reducible to LP and therefore IP would be a subset of P. IP is a subset of NP-Complete and as such, showing that IP is a subset of P would prove P = NP.

VIII. Moving Forward

The next steps for this approach to IP is to prove or disprove the unproven conjectures contained within this paper and continue research from there depending on the conslusion regarding said unproven conjectures.