Integer Linear Programming:

A Case Study on Tournament Scheduling

Contents

1.0 Introduction

Once again, I was sitting restlessly at an airport gate, waiting for my layover flight home. Every hour of the last three days had been consumed at a debate tournament, and I was working to finish several days' schoolwork in order to return to class the following day. However, after losing another close elimination round which lost me the chance to bid at that tournament, it was difficult to refocus my attention.

In many ways, national circuit debate is one of the most demanding competitive activities available to high school students. Debaters of the most successful tier compete for bids to the Tournament of Champions (ToC) by advancing deeply enough into elimination rounds at nationally respected tournaments, which are carefully chosen and allotted a certain number of bids based on its perceived competitiveness by the ToC committee prior to each season. The ToC is unanimously the most prestigious and competitive tournament of the year—the pinnacle of American high school debate, and takes place on the last week of April every year.

After repeated emotionally taxing failures, the simplest rejoinder seemed to be continuing to register for more tournaments. However, this kind of unrestrained traveling required heavy time investment and came at significant monetary expenses. Therefore, at the beginning of a debate season, the problem that I encounter is deciding which tournaments I should attend in order to maximize the number of bids I earn, given various constraints such as the maximum cost and time that can be spent on these tournaments. As there are usually around 40 ToC-qualifying tournaments each season, the number of combinations of tournaments I can attend is beyond the millions. There must be a more sensible way to determine the optimal solution.

After a little research, I learned of an active research area called mathematical optimization. My tournament scheduling could be formulated as an optimization problem, an

integer linear programming (IP) task in particular. It must have been a coincidence that among the heavy load of schoolwork on my hands, one of them was to choose a topic for my IB Mathematics Internal Assessment.

In this paper, I will describe the basic concepts and common algorithms for IP. I will start with *linear programming* (LP) in Section 2, and then move onto the IP in Section 3. I will use the tournament problem as an example to illustrate how these common algorithms work. All the algorithms covered in this paper were developed in the past century and have been covered in numerous books on mathematic optimization.

2.0 Linear Programming

Linear programming (LP) is a method to solve an optimization problem in which the objective function is linear and the constraints may be expressed as linear equalities or inequalities. LP is one of the most important and widely used optimization techniques. It is attractive both for its applicability to real-world situations, and its solvability through practically efficient techniques on large-scale problems.

In this section, we first formulate the tournament example into an LP problem, then describe the standard form and canonical form of LP, and finally discuss one common method for solving LP problems, called the simplex method.

2.1 The Tournament Problem

We develop our first example based on the question posed in the introduction of how to maximize bids to the Tournament of Champions, one that will be used repeatedly throughout the paper to develop various aspects of linear programming. For simplicity's sake, we will only consider ToC-qualifying tournaments with octafinals or quarterfinals bids, which tend to offer

the best competition. In the 2014-2015 season, there are 20 such tournaments, each of which we assigned a number based on their chronological order for mathematical organization. The data for these tournaments is represented below.

Tournament	**Assigned Number (i)**	**Chance of bidding (c)**[1]	**Cost (m)**[2]	**Time needed (t)**[3]	**Date**[4]
Greenhill	1	0.4	$500	3	9/20-9/21
Yale	2	0.3	$700	4	9/19-9/21
Valley	3	0.5	$500	3	9/27-9/29
Cypress Bay*	4	0.4	$600	4	10/10-10/12
Presentation	5	0.5	$350	3	10/11-10/13
Bronx Science	6	0.3	$600	3	10/17-10/19
St. Mark's	7	0.5	$500	3	10/18-10/20
Meadows	8	0.6	$450	4	10/31-11/2
Apple Valley*	9	0.3	$300	2	11/7-11/8
Glenbrooks*	10	0.5	$400	3	11/22-11/24
Alta*	11	0.4	$500	4	12/4-12/6
Blake	12	0.5	$450	4	12/19-12/21
College Prep	13	0.4	$350	3	12/20-12/21
Harvard Westlake	14	0.5	$400	3	1/3-1/5
Sunvitational	15	0.3	$600	3	1/11-1/12
Lexington	16	0.4	$500	3	1/17-1/18
Emory	17	0.5	$600	4	1/23-1/25
Stanford	18	0.6	$350	4	2/7-2/9
Berkeley	19	0.4	$350	3	2/14-2/16
Harvard	20	0.5	$500	4	2/14-2/16

Table 1: The debate tournaments in the 2014-2015 season

[1] At my skill level, I assume that my chance of earning a bid at a tournament is between 30% and 60%, depending on factors such as historic difficulty, tournament style, etc.
[2] The total cost of a tournament is the sum of plane tickets, tournament registration, and miscellaneous expenses.
[3] The number of school days missed for a tournament is determined through consideration of the tournament date, my school schedule, and whether long flights will require me to miss additional school days.
[4] Tournaments that occur on the same weekend are highlighted in the same color. This will be discussed later in the paper.

The first step is to define the *decision variables*, which describe the choice under our control. As we need to make a decision for each tournament, each one needs its own decision variable, written as

$$x_i, \quad i = 1, 2, \ldots, 20.$$

For each tournament under consideration, the question raised is whether or not to attend it. This means the decision variable for each tournament must have a binary value—either I go, or I don't. Thus,

(1) $$x_i = \begin{cases} 1 & \text{if decided to compete at the } i^{\text{th}} \text{ tournament} \\ 0 & \text{otherwise} \end{cases}$$

The second step is to define the objective function. In our problem, the goal of competing is to maximize bids. Let c_i represent the chance of earning a bid at the i^{th} tournament. The *objective function* z is then defined as

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

The final step is to represent the *constraints*, limitations that restrict our choices for decision variables. Of course, flying around the nation to compete is quite expensive, so we must work within the total budget, *M*, that my family can afford. Likewise, there is a limit to how many days I can spend away from home and school, *T*, if I wish to stay caught up in school. Furthermore, the values of decision variables have to be 1 or 0. Lastly, I obviously cannot be at two tournaments on the same weekend (e.g., the first two tournaments in Table 1). This constraint can be presented as $x_i + x_j \leq 1$ if the i^{th} and j^{th} tournaments have time conflict; that is, at most one of the two decision variables can have value 1. Together, these constraints can be expressed as

(2)
$$\sum_{i=1}^{n} m_i x_i \leq M$$

$$\sum_{i=1}^{n} t_i x_i \leq T$$

$$x_i \text{ is 0 or 1 for each } i$$

$$x_1 + x_2, x_4 + x_5, x_6 + x_7, x_{12} + x_{13}, x_{19} + x_{20} \leq 1$$

Hence, the optimization problem we shall study can be formulated as follows:

(3)
$$\text{maximize z} = \sum_{i=1}^{n} c_i x_i$$

$$\text{subject to} \sum_{i=1}^{n} m_i x_i \leq M$$

$$\sum_{i=1}^{n} t_i x_i \leq T$$

$$0 \leq x_i \leq 1 \text{ for every } i$$

$$x_i \text{ is an integer for every } i$$

$$x_1 + x_2 \leq 1$$

$$x_4 + x_5 \leq 1$$

$$x_6 + x_7 \leq 1$$

$$x_{12} + x_{13} \leq 1$$

$$x_{19} + x_{20} \leq 1$$

In this problem, the objective function is linear. All the constraints except the fourth one, which limits x_i to an integer, are linear inequalities. If we ignore that constraint for the time being, the problem is an LP problem, and we can apply LP techniques to it.

To demonstrate how the LP algorithms work, let us simplify the example. Instead of using all 20 tournaments, suppose we have narrowed down the choices to the four tournaments marked with the asterisk in the first column in Table 1. Since there are no time conflicts among them, we can ignore the tournament dates in the last column. We also renumber the tournaments. The new table is below.

Tournament	**Assigned Number (i)**	**Chance of bidding (c)**	**Cost (m)**	**Time needed (t)**	**Date**
Cypress Bay	1	0.4	\$600	4	10/10-10/12
Apple Valley	2	0.3	\$300	2	11/7-11/8
Glenbrooks	3	0.5	\$400	3	11/22-11/24
Alta	4	0.4	\$500	4	12/4-12/6

Table 2: A simplified task, where four of the 20 tournaments in Table 1 are chosen

Suppose the budget for travel expense, *M*, is \$1500, and the maximum number of days allowed for competing, *T*, is 10 days. We substitute the coefficients in the LP problem with these and the numbers in Table 2, and the new problem becomes:

$$
\begin{aligned}
(4) \qquad & \text{maximize } z = 0.4x_1 + 0.3x_2 + 0.5x_3 + 0.4x_4 \\
& \text{subject to } 600x_1 + 300x_2 + 400x_3 + 500x_4 \le 1500 \\
& 4x_1 + 2x_2 + 3x_3 + 4x_4 \le 10 \\
& 0 \le x_i \le 1, \qquad i = 1, 2, 3, 4 \\
& x_i \text{ is an integer}, \qquad i = 1, 2, 3, 4
\end{aligned}
$$

To simplify the task, we multiple the objective function by 10 to get rid of decimals, and divide both sides of the first constraint by 100. The new problem below will have the same optimal solution as the original one in (4):

(5) $$\text{maximize } z = 4x_1 + 3x_2 + 5x_3 + 4x_4$$

$$\text{subject to } 6x_1 + 3x_2 + 4x_3 + 5x_4 \leq 15$$

$$4x_1 + 2x_2 + 3x_3 + 4x_4 \leq 10$$

$$0 \leq x_i \leq 1, \qquad i = 1, 2, 3, 4$$

$$x_i \text{ is an integer}, \qquad i = 1, 2, 3, 4$$

2.2 Properties of Linear Programming

The simplex method is one of the most well-known methods to solving an LP problem. First, we define some terms. Given an optimization problem, a *solution* is the assignment of the values to all the decision variables. A solution is called *feasible* if it satisfies all the constraints. The set of all the feasible solutions form a *feasible region*. A feasible solution is called *optimal* if the objective function attains the optimal value at the solution. If a problem has no feasible solution, the problem itself is called *infeasible*, as the following example illustrates:

(6) $$\text{maximize z} = x_1 + 2x_2$$

$$\text{subject to } x_1 + x_2 \leq 2$$

$$-2x_1 - 2x_2 \leq -9$$

$$x_1, x_2 \geq 0$$

In this example, the first constraint requires $x_1 + x_2 \leq 2$, and the second constraint requires $x_1 + x_2 \geq 4.5$ (simplified through dividing both sides of the second inequality by -2). Clearly, no (x_1, x_2) pair can meet both requirements, and the problem is therefore infeasible.

If the value of the objective function can be infinitely large, we call the problem *unbounded*. For example, consider the following problem:

(7)
$$\text{maximize } z = x_1 - 4x_2$$
$$\text{subject to } -4x_1 + x_2 \leq -1$$
$$-x_1 - 2x_2 \leq -3$$
$$x_1, x_2 \geq 0$$

Here, we could set x_2 to zero and any x_1 larger than 3 with that x_2 will meet all the constraints. Therefore, x_1 can be infinitely large, and so can the value of z. The problem is thus unbounded.

An LP problem has some nice properties. First, the feasible region of an LP problem is always convex. A set is called *convex* if for every two points in the set, the line segment joining the points is also in the set. In a convex set, a *corner point* of a feasible region is a point that is not the midpoint of two other points of the feasible region. To understand what a corner point is, let us look at the LP problem when there are only two decision variables. Let us make the two variables correspond to the two axes in a two-dimension space. Now each equality constraint is a line that divides the space into two halves, and the feasible region is the polygon formed by those lines. In Figure 1, the shaded region is the feasible region of the LP problem in (8). The feasible region is formed by the *x* –axis, the *y*-axis, $x_1 + x_2 \leq 2$ on top, and $8x_1 + x_2 \leq 4$ on the right. The corner points are the vertices of the polygon. In other words, these corner points happen at the intersection of two constraints. The polygon is convex because if we connect any two points inside the polygon, the line segment is also inside the polygon. If an LP problem has three decision variables, the feasible region is a polyhedron; the corner points are vertices of the polyhedron and they happen at the intersection of three constraints.

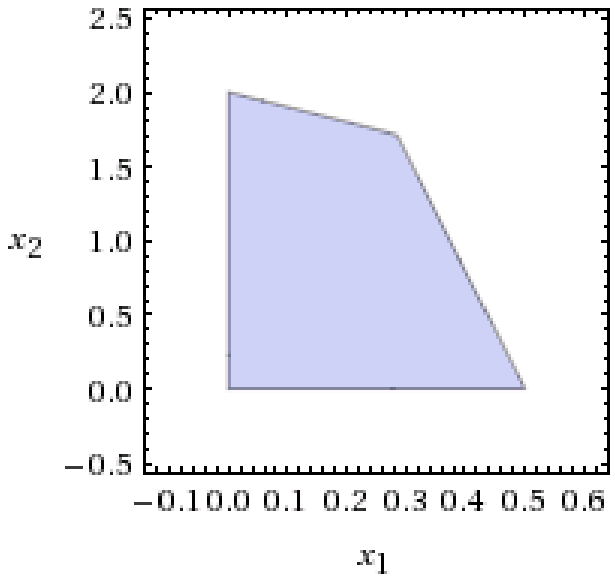

Figure 1: Feasible Region of LP Problem in (8)

(8)
$$\text{maximize } z = x_1 - 3x_2$$
$$\text{subject to } x_1 + x_2 \leq 2$$
$$8x_1 + x_2 \leq 4$$
$$x_1, x_2 \geq 0$$

For the LP problem, the following statements are true:

1) If every decision variable is non-negative and if the feasible region is non-empty, then the region has a corner point.
2) If the feasible region is non-empty and bounded, then the problem has an optimal solution.
3) If the problem has an optimal solution and its feasible region has a corner point, then there is an optimal solution at that corner point.

Combining all three statements, we arrive at the following conclusion: if every decision variable of an LP problem is non-negative and the feasible region is non-empty and bounded, the

problem has an optimal solution (statement 2), the feasible region has a corner point (statement 1), and there is an optimal solution at a corner point (statement 3). Therefore, to find the optimal solution for a feasible, bounded LP problem with non-negative variables, we just need to check all the corner points of the feasible region and choose the one with the optimal objective function value. That is the intuition behind the simplex method, which we will discuss in Section 2.5.

2.3 Standard Form

Before trying to solve an LP problem, let us first define the standard form for LP. A linear program is in standard form if the following are all true: 1) all decision variables are non-negative, 2) all constraints are expressed as linear equalities, and 3) the constants on the right-hand-side of linear equalities are non-negative. Without the loss of generality, let us also assume that the task is always to maximize the objective function, not to minimize it. Thus, the standard form has the following form:

(9)
$$\text{maximize } z = \sum_{j=1}^{n} c_j \, x_j$$

$$\text{subject to } \sum_{j=1}^{n} a_{ij} x_j = b_i, \qquad i = 1, 2, \dots, m$$

$$x_j \geq 0, \qquad j = 1, 2, \dots, n$$

$$b_i \geq 0, \qquad i = 1, 2, \dots, m$$

Any LP problem can be converted to the standard form. For instance, if the LP problem includes an inequality

$$a_1 x_1 + a_2 x_2 + \cdots + a_n x_n \leq b$$

we can introduce an equality by adding a new nonnegative variable *w*:

$$a_1 x_1 + a_2 x_2 + \cdots + a_n x_n + w = b$$

We call *w* a slack variable, as it fills the gap between the left-hand side and the right-hand side of the inequality. It is easy to prove that the new LP problem with this new slack variable and the equality replacing the inequality is equivalent to the original LP problem. That is, the two problems will have the same optimal values. Similarly, the opposite inequality constraint

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n \geq b$$

can be converted to an equality by adding a surplus non-negative variable *w*:

$$a_1x_1 + a_2x_2 + \cdots + a_nx_n - w = b$$

For the second requirement of the standard form, if a variable x_j is negative, we can simply define a new variable $y_j = -x_j$, and replace x_j with $-y_j$ in both the objective function and all the constraints. For the third requirement, if a particular b_i is negative, we can simply multiply both sides of the equality constraint by -1, and the new right-hand side will now be non-negative.

Now let us go back to our tournament example with the integral constraint removed, repeated below.

(10)
$$\text{maximize } z = 4x_1 + 3x_2 + 5x_3 + 4x_4$$
$$\text{subject to } 6x_1 + 3x_2 + 4x_3 + 5x_4 \leq 15$$
$$4x_1 + 2x_2 + 3x_3 + 4x_4 \leq 10$$
$$0 \leq x_i \leq 1, \qquad i = 1, 2, 3, 4$$

It is not in the standard form, as it includes two inequality constraints for the budget and time allowed, and four inequalities requiring that $x_i \leq 1$ for each x_i. Therefore, we need to introduce six slack variables, $x_5, x_6, \ldots, x_{10}$, one for each inequality. The resulting equivalent LP problem in standard form is below:

(11)
$$\text{maximize } z = 4x_1 + 3x_2 + 5x_3 + 4x_4$$
$$\text{subject to } 6x_1 + 3x_2 + 4x_3 + 5x_4 + x_5 = 15$$
$$4x_1 + 2x_2 + 3x_3 + 4x_4 + x_6 = 10$$
$$x_1 + x_7 = 1$$
$$x_2 + x_8 = 1$$
$$x_3 + x_9 = 1$$
$$x_4 + x_{10} = 1$$
$$0 \leq x_i, \quad i = 1, 2, 3, 4$$

2.4 Canonical Form

In the standard form, suppose there are *n* decision variables, *m* equality constraints, and $m < n$. In our tournament example, *n* is 10, and *m* is 6. The *basic* variables are the variables appearing in exactly one equality constraint. The *non-basic* variables are all the remaining variables. The *canonical form* of LP is the standard form plus an additional requirement that each equation constraint has exactly one basic variable with coefficient 1, and the objective function includes only non-basic variables. The LP problem in Equation (11) is already in the canonical form as each of the slack variables, $x_5, x_6, \ldots, x_{10}$, appears in exactly one equality, and they are basic variables; each equality contains exactly one basic variables with coefficient 1, and the objective function includes only non-basic variables.

If a standard form is not in the canonical form, we can convert it to the canonical form by using the pivot operation. The pivot operation is the same operation as used in Gaussian elimination to solve linear equation, except that the number of constraints in an LP problem is not the same as the number of the variables. The idea is that we first choose *m* variables as the

basic variables. Then for each basic variable, keep it in one equality constraint, and use that to eliminate the variable from other constraints that containing the variables.

2.5 The Simplex Method

Recall that at the end of Section 2.2, we mentioned that, to find the optimal solution for a feasible, bounded LP problem with non-negative variables, we just need to check all the corner points of the feasible region and choose the one with the optimal objective function value. The main idea of the simplex method is to start at a feasible corner point, find an edge in which the objective value is improving, and follow the edge to go to the next corner point. Continue the process until no adjacent corner point has a better objective value. To be more specific, the algorithm has three steps:

1. Convert the LP problem into the standard form, as explained in Section 2.3.
2. Convert the standard form into canonical form, and find an initial basic feasible solution
3. While the objective function z has a positive co-efficient
    a. Choose a non-basic variable x_j whose co-efficient in z is positive. We call x_j the *entering variable*.
    b. For each current basic variable x_k, calculate $\frac{b_k}{a_{k,j}}$, where b_k is the right-hand side of the only constraint that x_k appears in, and $a_{k,j}$ be the co-efficient of x_j in that equality. Let x_i be the basic variable with the minimal value of $\frac{b_k}{a_{k,j}}$ among all the x_k $whose$ $a_{k,j} > 0$; that is, $i = \arg min_k(\frac{b_k}{a_{k,j}}, where\ a_{k,j} > 0)$. We call x_i the *exiting variable*.

c. Get the LP problem into the new canonical form where x_j replaces x_i as the basic variable, and x_i becomes a non-basic variable. All other variables remain the same. Now we arrive at a new basic feasible solution.

4. Once we exit the loop in Step 3, the last basic feasible solution is the optimal solution.

To understand the algorithm, we first need to define the concept of basic solution. Given an LP problem in canonical form with a specific set of basic variables, the associated *basic solution* is the unique solution obtained by setting the non-basic variables to 0. Because a basic variable appears in exactly one equality and each equality includes exactly one basic variable, once we set the non-basic variables to 0, the value of a basic variable x_i must be equal to the b_i. If the solution is feasible (i.e., the b_i in the canonical form are non-negative), it is called a *basic feasible solution*. A basic feasible solution is a corner point solution. Therefore, in Step 3, we move from one corner point to another corner point, and the second corner point yields a higher objective value.

Let us demonstrate how the algorithm works with our toy example in (11), repeated here as (12).

(12)
$$\begin{aligned} &\text{maximize } z = 4x_1 + 3x_2 + 5x_3 + 4x_4 \\ &\text{subject to } 6x_1 + 3x_2 + 4x_3 + 5x_4 + x_5 = 15 \\ &\qquad 4x_1 + 2x_2 + 3x_3 + 4x_4 + x_6 = 10 \\ &\qquad x_1 + x_7 = 1 \\ &\qquad x_2 + x_8 = 1 \\ &\qquad x_3 + x_9 = 1 \\ &\qquad x_4 + x_{10} = 1 \\ &\qquad 0 \leq x_i, \;\; i = 1, 2, 3, 4 \end{aligned}$$

The form in (12) is already in the canonical form with $x_{5,}x_{6,}\ldots,x_{10}$ as the basic variables. Once we set non-basic variables, $x_{1,}\ x_{2,}\ x_{3,}\ and\ x_{4,}$ to be zero, the basic variables must have value 1, and the current objective value is 0. In other words, $(0,0,0,0,15,10,1,1,1,1)$ is our initial basic feasible solution in Step 2. This solution means that I choose to go to none of the tournaments and the objective value is 0. In that case, I will have 15 ($1500) left for the budget and 10 days unused for time allocation.

In Round 1 of Step 1 (i.e., the first time we enter Step 3), all four coefficients of non-basic variables in z are positive. While we can choose any of them as entering variables, suppose we choose x_1. We want to increase the value of x_1 to be as high as possible because a higher value of x_1 yields a higher value of z, as shown by the objective function. On the other hand, we want to ensure that all the constraints are still satisfied. The variable x_1 appears in the first three constraints, and $min\left(\frac{b_k}{a_{k,1}}\right)=1$ for $k=1,2,3$, and $argmin\left(\frac{b_k}{a_{k,1}}\right)=7$; that is, the exiting variable is x_7 because the third equality $x_1+x_7=1$ sets the highest value x_1 can be. So now the new set of basic variables are $x_{1,}x_{5,}x_{6,}\ x_{8,}x_{9,},x_{10}$. The corresponding basic solution is $(1,0,0,0,9,6,0,1,1,1)$; which means we decide to go to only the first tournament, and the unused budget is 9 ($900) and the unused days are 6. The objective value is 4. Now in Step (3c), we convert (12) into canonical form with this new set of basic variables, as shown in (13).

(13)
$$\text{maximize } z=4+3x_2+5x_3+4x_4-4x_7$$
$$\text{subject to } 3x_2+4x_3+5x_4+x_5-\ 6x_7=9$$
$$2x_2+3x_3+4x_4+x_6-\ 4x_7=6$$
$$x_1+x_7=1$$
$$x_2+x_8=1$$
$$x_3+x_9=1$$

$$x_4 + x_{10} = 1$$

$$0 \leq x_i, \quad i = 1, 2, 3, 4$$

Notice that in (13), the last four constraints remain the same. The first two look different because we want to ensure x_1 as a basic variable only appears in one equality constraint. The z function looks also different as we replace $4x_1$ in (12) with $4 - 4x_7$ in (13).

In Round 2 of Step 3, we can choose x_2 as the entering variable and the corresponding exiting variable will be x_8. The new basic solution will be $(1, 1, 0, 0, 6, 4, 0, 0, 1, 1)$, meaning we decide to go to the first two tournaments, and the unused budget and days will be 6 and 4, respectively. The objective value is 7, and the new canonical form will be (14).

(14)
$$\text{maximize } z = 7 + 5x_3 + 4x_4 - 4x_7 - 3x_8$$

$$\text{subject to } 4x_3 + 5x_4 + x_5 - 6x_7 - 3x_8 = 6$$

$$3x_3 + 4x_4 + x_6 - 4x_7 - 2x_8 = 4$$

$$x_1 + x_7 = 1$$

$$x_2 + x_8 = 1$$

$$x_3 + x_9 = 1$$

$$x_4 + x_{10} = 1$$

$$0 \leq x_i, \quad i = 1, 2, 3, 4$$

In Round 3 of Step 3, we can choose x_3 as the entering variable and the corresponding exiting variable will be x_9. The new basic solution will be (1, 1, 1, 0, 2, 1, 0, 0, 0, 1), meaning we decide to go to the first three tournaments, and the unused budget and days will be 2 and 1, respectively. The objective value is 12, and the new canonical form will be (15).

(15)
$$\text{maximize } z = 12 + 4x_4 - 4x_7 - 3x_8 - 5x_9$$
$$\text{subject to } 5x_4 + x_5 - 6x_7 - 3x_8 - 4x_9 = 2$$
$$4x_4 + x_6 - 4x_7 - 2x_8 - 3x_9 = 1$$
$$x_1 + x_7 = 1$$
$$x_2 + x_8 = 1$$
$$x_3 + x_9 = 1$$
$$x_4 + x_{10} = 1$$
$$0 \leq x_i, \quad i = 1, 2, 3, 4$$

So far, the three basic solutions have been very intuitive. Each time, we decide to go to one more tournament; as a result, the unused money and days decreases and the objective value (which is equal to 10 times the bids I expect to get) increases. Now in Round 4, we can choose x_4 as the entering variable and the corresponding exiting variable will be x_6. The new basic solution will be $(1, 1, 1, 0.25, 0.75, 0, 0, 0, 0, 0.75)$, meaning we decide to go to the first three tournaments and go to the fourth tournament 0.25 times, and the unused budget and days will be 0.75 and 0, respectively. The objective value is 13, and the new canonical form will be (16).

(16)
$$\text{maximize } z = 13 - x_6 - 4x_7 - 3x_8 - 5x_9$$
$$\text{subject to } x_4 + 0.25x_6 - x_7 - 0.5x_8 - 0.75x_9 = 0.25$$
$$x_5 - 1.25x_6 - 11x_7 - 5.5x_8 - 7.75x_9 = 0.75$$
$$-0.25x_6 + x_7 + 0.5x_8 + 0.75x_9 + x_{10} = 0.75$$
$$x_1 + x_7 = 1$$
$$x_2 + x_8 = 1$$
$$x_3 + x_9 = 1$$
$$0 \leq x_i, \quad i = 1, 2, 3, 4$$

Notice that in (16) all the coefficients of variables in the objective function are negative, which means that we will exit the while loop in Step 3, and the current basic solution is the optimal solution. In other words, our optimal solution is to go to each of the first three tournaments once, and the last tournament 0.25 times, and the objective value is 13. (i.e., 1.3 bids). That arrangement will use up all 10 days, and have 0.75 budget (i.e., $75) left. However, what does it mean to go to a tournament 0.25 times? Recall that in order to treat this task as an LP problem, we ignored the constraint that each decision variable must be an integer. How do we deal with fractional values? That is the topic of the next section.

3.0 Integer Programming

An integer programming (IP) problem is an LP problem with the additional constraint that some or all of the decision variables must be integer valued. It can be divided into three types:

- Mixed IP: some of the decision variables must be integer valued.
- Pure IP: all the decision variables must be integer values.
- 0-1 IP: all the decision variables are binary; that is, their values are 0 or 1.

Our tournament scheduling model belongs to the last type, as all the decision variables in are binary, indicating whether I choose to go to that tournament.

IP is an active research area, and there are still many unsolved problems. One venue to find out recent progress on IP is to attend IPCO (Integer Programming and Combinatorial Optimization), an international conference sponsored by the Mathematical Optimization Society and is held twice in every three-year period since 1990.[5] In this section, we just provide a brief

[5] http://www.mathopt.org/?nav=ipco

introduction to IP, point out its relevance to our debate example, and outline two common methods to solve IP problems.

3.1 Why Integer Programming?

Integer programming is important for several reasons. First, many real-life applications, including our debate scheduling model, require that the decision variables are integers. Second, IP allows us to model certain logical constraints easily. For instance, we can use $x_1 + x_2 \leq 1$ to represent the constraint that "if I go to the first tournament, then I cannot go to the second one," and use $x_1 \leq x_2$ to represent the constraint that "if I go to the first tournament, then I must also go to the second one." Third, IP allows us to model certain non-linear objective function. One such an example is piecewise linear function, i.e., a function with several pieces and each piece is linear.[6] Finally, IP can model almost any combinatorial optimization problem, including Traveling Salesman Problem, Set Packing Problem, Set covering problem, Graph Coloring Problem, and many more.[7]

While IP has many advantages as mentioned above, its disadvantage is that, compared to linear programming (LP) problems, IP is more difficult to model and can be more difficult to solve. For instance, many IP problems (e.g., Traveling Salesman Problem) are NP-complete, meaning that it is unlikely that the problems can be solved in polynomial time.

How can we solve an IP problem? If we are lucky, we can simply round up the LP solution. In our debate example, the values of x_1, x_2, x_3, x_4 are 1, 1, 1, 0.25 respectively. If we

[6] An example of modeling piecewise linear function is given in slides #25-27 at http://ocw.mit.edu/courses/sloan-school-of-management/15-053-optimization-methods-in-management-science-spring-2013/lecture-notes/MIT15_053S13_lec11.pdf

[7] http://ocw.mit.edu/courses/sloan-school-of-management/15-053-optimization-methods-in-management-science-spring-2013/lecture-notes/MIT15_053S13_lec14.pdf

round them to 1, 1, 1, and 0, the solution is indeed a feasible solution, and an optimal solution, to the IP problem. But more often, the rounding-up approach will not work. To address this issue, various methods are proposed to speed up the process; two of them are discussed in the next section.

3.2 Branch and Bound and cutting planes

The first method is called *Branch and Bound (B&B)*, developed because enumerating all possible solutions is too slow. For instance, when there are 20 tournaments, the number of all possible solution is 2^{20}, which is a little bit over 1 million. B&B uses an enumeration tree and prune the tree heavily, taking advantage of the LP relaxation of the problem.

Given an IP problem k, if we drop the requirements that variables be integer, we call the new problem *the LP relaxation* of the original IP problem. Let us use IP(k) to denote the original problem, and LP(k) to denote the LP relaxation. If the optimal solution for LP(k) is feasible for IP(k), meaning that the values for decision variables are all integers, the solution is also optimal for IP(k). If LP(k) is infeasible, so is IP(k). Quite often, finding the optimal solution for LP(k) is much faster than finding the optimal solution for IP(k).

B&B builds an enumeration tree. The root is the original IP problem. Let us call it IP(1). Next, we choose a decision variable, say x_1. Suppose the variable is binary, now we create two children for the root, one for each possible value of x_1. Each child represents a new IP problem with the additional constraint that x_1 has that value. Let us call the children IP(2) and IP(3). Now we expand each child by choosing another decision variable. Repeat the process until all the leaf nodes cover all possible solutions for the original problem. As you can imagine, the enumeration tree is huge.

The trick of B&B is to keep track of the best solution so far in the B&B search and prune the enumeration tree accordingly. We call the current best solution *the incumbent*. Now for any node k in the tree that we have not examined, we solve LP(k) using linear programming. There are three cases:

1. If the LP(k) solution is worse than the incumbent (i.e., the LP(k) objective value is lower than the objective value attained at the incumbent), we know that the IP(k) solution will be even worse because the objective value of IP(k) is always no larger than that of LP(k). Therefore, there will be no need to calculate IP(k) and we will prune the subtree rooted at node k.
2. If the LP(k) solution is better than the incumbent and the solution is also feasible for IP(k), we know the solution is also optimal for IP(k). Now we can treat that solution as our new incumbent and prune the subtree at k.
3. If the LP(k) solution is better than the incumbent but the solution is not feasible for IP(k), the IP(k) solution could be better or worse than the incumbent. Therefore, we need to check the children of k.

The B&B search ends when all the nodes in the enumeration tree have been either checked or pruned. In essence, the B&B algorithm speeds up the process of solving the IP problem by comparing the LP solution with the incumbent and pruning the enumeration tree accordingly.

The second IP method is called *cutting planes*. The idea is to iteratively reduce the feasible region of the corresponding LP problem by introducing *valid inequalities*. A valid inequality for an IP is any constraint that does not eliminate any feasible integer solutions. The intuition is that let us first find an optimal solution to the LP problem without the integer constraints. If the solution is not feasible to the IP problem, we will introduce valid inequalities to reduce the

feasible region for the LP problem so that the current optimal solution is no longer inside the feasible region of the new LP problem. Now repeat the process until we find an optimal LP solution that meets the integer constraints. That solution is therefore the optimal solution to the IP problem.

4.0 Conclusion

It is incredible that something as seemingly difficult to grasp as the secret behind earning bids can be modeled by mathematics. Nonetheless, the real-life situation is much more complex than the task we laid out in Section 2.1. For instance, we assume that c_i , the chance of earning a bid at a tournament, is known and will not change with respect to the values of the decision variables. In reality, the chance of getting a bid at the i^{th} tournament can be influenced by the number of tournaments that I have competed at before then, as skill is built off experience. Another issue lies in the objective function. Right now, the sole objective is to maximize the number of bids I earn. In reality, if there is more than one optimal solution, I will prefer the one that costs less or takes fewer days. In our model, there are two optimal solutions. One is to go to the first three tournaments, and the other is to go to the last three. Both will result in 1.2 bids and take 9 days, but the second option costs $100 less. To take the cost and time into consideration, we would need to change the objective function.

Finally, there are factors that I am unaware of when deciding whether to register. After taking a costly redeye flight from Seattle to Boston to attend the Harvard tournament, a powerful snowstorm forced partial cancellation of the tournament, and its bid status was rescinded by the ToC committee. My chances of earning a bid crashed from 0.5 to 0, and once more I am stuck at an airport hoping to catch a flight back home, this time bringing my research to a conclusion.

Bibliography

Bradley, Stephen P., Arnoldo C. Hax, and Thomas L. Magnanti. *Applied Mathematical Programming*. Reading, MA: Addison-Wesley Pub., 1997. Print.

Land, A.H., and A.G. Doig. *An Automatic Method of Solving Discrete Programming Problems.* London: London School of Economics and Political Science, 1960. Print.

Thie, Paul R., and G.E. Keough. *An Introduction to Linear Programming and Game Theory*. Hoboken, NJ. J. Wiley & Sons, 2008. Print.

Wolfram Research Inc. Wolfram Mathematica. Computer software. Wolfram Mathematica. Vers. 8.0. Wolfram Research Inc., 15 Nov. 2010. Web. 22 Feb. 2014.

Vanderbei, Robert J. *Linear Programming: Foundations and Extensions.* Boston: Kluwer Academic, 2001. Print.

Zionts, Stanley. *Linear and Integer Programming.* Englewood Cliffs, NJ: Prentice-Hall, 1974. Print.