7 Beyond LP Relaxation

LP relaxation is a powerful tool for approximating, or sometimes exactly finding, the solution of an ILP, but as we have already seen, rounding the non-integral LP solution is generally difficult. In this part, we discuss some methods to enhance the LP relaxation for integer programming. We will talk about the **primal-dual**, **branch-and-bound** and **cutting plane** techniques. From these three, I will discuss only the first one in details. The two others will only be briefly introduced. Hence, I do not expect you to be able to implement the branch and bound or the cutting plane algorithms, but you should be able to elaborate on the general ideas behind them. An implementation of these techniques can be found in the MATLAB function `intlinprog`. I remind that the ILP problems are NP-complete, hence we do not expect any of these techniques to exactly solve the generic problem. For example, the MATLAB function `intlinprog` has an adjustable time limit. If the solution is not found before this deadline, the program will return the best solution found so far.

Let us start by the primal-dual technique. The primal-dual method is in fact a generic method for solving convex optimization problems. However, the original technique generally behaves poorly, especially for the LP problems. There are a number of modified primal-dual techniques, including some interior point methods, which have a state-of-the-art performance.

Although the primal-dual technique is not as suitable as, for example, the simplex method for LP problems, it can help instead in some ILP ones to design approximate algorithms. To explain this approach, I start from the basic technique for LP problems. Remember that a pair $(\mathbf{x}, \mathbf{u})$ of primal and dual feasible solutions is an optimal pair for both the primal and dual problems if and only if this pair satisfies the complementary slackness conditions. This means that if the conditions are not satisfied, one can improve the primal and dual costs by modifying the pair. We are going to explain the details on the standard form in 37, which we more often encounter.

There are many specific approaches to the primal-dual algorithm. The one that we consider here is a simplified version, which is related to the Lagrange method of multipliers and can be explained as follows:

1. Start by a dual feasible solution $\mathbf{u}_0$ (In many cases, $\mathbf{u}_0 = \mathbf{0}$ should work). Set the iteration number t to 0.

2. At each iteration, we will find all (possibly infeasible) primal variable $\mathbf{x}$ that satisfy the primal part of the complementary slackness conditions. This means that we find the non-active dual constraints at $\mathbf{u}_t$ and set their corresponding primal variables to zero. The others can be arbitrary (but positive). Suppose that $j_1, j_2, \ldots, j_p$ are the index of those primal variables, whose corresponding dual constraints are inactive, and hence are arbitrarily positive.

3. Now, our goal is to update one of the dual variables, say u_i to $u_i+\epsilon$, such that the constraints are still satisfied, the dual cost is increased and in a sense we move toward satisfying the entire complementary slackness conditions. For this, we look for an index i and a nonzero real number ϵ, such that updating u_i to $u_i+\epsilon$ and keeping the other dual variables unchanged leads to a dual feasible solution, and one of the two following situations happen:

   - The step value ϵ is negative: The i^{th} primal constraint is always violated, no matter how we choose $x_{j_1}, x_{j_2}, \ldots, x_{j_p} \geq 0$, while the other primal variables are zero.
   - The step value ϵ is positive: The i^{th} primal constraint is always satisfied and inactive, no matter how we choose $x_{j_1}, x_{j_2}, \ldots, x_{j_p} \geq 0$, while the other primal variables are zero.

4. If such an index i and step ϵ exist, add ϵ to the i^{th} entry of $\mathbf{u}_t$ to obtain $\mathbf{u}_{t+1}$, update the iteration number t to $t+1$ and go to step 2. Otherwise, stop.

5. Now, $\mathbf{u}_t$ is the dual optimal solution. Find the primal solution $\mathbf{x}$ from the complementary slackness conditions.

The algorithm scheme above may sound complicated and counterintuitive. Indeed, I have skipped many details regarding the derivation of the scheme, but the next example shows that in fact the scheme often leads to intuitive algorithms. Another fact, which we encounter in the next example is that, the primal-dual method can be easily used by restricting the primal values to be integer (or binary in the case of 0-1 integer programing). Then, the result is always integral.

Example 21. The Internet had been expanding rapidly in the Free Republic of West Mordor, and the government issued a regulation, purely in the interest of improved security of the citizens, that every data link connecting two computers must be equipped with a special device for gathering statistical data about the traffic. An operator of a part of the network has to attach the government's monitoring boxes to some of his computers so that each link has a monitored computer on at least one end. Which computers should get boxes so that the total price is minimum? Let us assume that there is a flat rate per box. Then, the number of boxes should be equivalently minimized. Such a set of computers is called a **minimum covering set** for the links.

Let us denote the computers in the network by $V = \{v_1, v_2, \ldots, v_n\}$ and the links by $E = \{e_1, e_2, \ldots, e_m\}$. Every link $e \in E$ is between two computers $v, w \in V$. We are to select a subset $S \subseteq V$ such that for every link $e \in E$, one of its ends is in S and S has the smallest possible cardinality. Introduce the indicator variables $x_i \in \{0, 1\}$, where $x_i = 1$ means that $v_i \in S$ and $x_i = 0$ implies that $v_i \notin S$. The cardinality of S equals $x_1 + x_2 + \ldots + x_n$. Hence, we can write the following optimization

$$\begin{aligned} &\min_{\mathbf{x}=(x_1,x_2,\ldots,x_n)\in\mathbb{Z}^n} \sum_{i=1}^{n} x_i \\ &\text{subject to} \\ &\forall e \in E, \quad \sum_{i|v_i\in e} x_i \geq 1 \\ &x_i \geq 0 \quad i = 1, 2, \ldots, n \end{aligned} \tag{66}$$

In the above, $v \in e$ means that v is connected to e. The first thing to observe is that I did not restrict the variables to be less than or equal to 1. The reason is that the solution to this optimization always consists of 0s and 1s. Suppose that one element, say x_1 is 2 for example. I can modify this variable to 1, without violating any constraint, while the cost will decrease. Now, let us take the LP relaxation:

$$\begin{aligned} &\min_{\mathbf{x}=(x_1,x_2,\ldots,x_n)\in\mathbb{R}^n} \sum_{i=1}^{n} x_i \\ &\text{subject to} \\ &\forall e \in E, \quad \sum_{i|v_i\in e} x_i \geq 1 \\ &x_i \geq 0 \quad i = 1, 2, \ldots, n \end{aligned} \tag{67}$$

The dual to this optimization can be written as

$$\begin{aligned} &\max_{\mathbf{x}=(u_1,u_2,\ldots,u_n)\in\mathbb{R}^n} \sum_{i=1}^{n} u_i \\ &\text{subject to} \\ &\forall v \in V, \quad \sum_{j|v\in e_j} u_j \leq 1 \\ &u_i \geq 0 \quad i = 1, 2, \ldots, m \end{aligned} \tag{68}$$

where each dual variable is related to one link (i.e., one constraint in the primal optimization). Let us try the primal-dual algorithm explained above.

1. We from $\mathbf{y}_0 = \mathbf{0}$ and set $t = 0$.

2. At each iteration, we find the indexes $I_t = \{i_1, i_2, \ldots, i_p\}$, for which $\sum_{i=1}^{n} u_i = 1$.

3. Now, we find a link j(equivalent to a dual variable u_j) and a step length ϵ such that

   - *Either* the j^{th} constraint is violated, no matter how I choose x_is for $i \in I_t$. It is simple to see that this is the case if and only if non of the two ends of this link is in I_t,
   - *or* the j^{th} constraint cannot get active. You may see that this cannot happen since at least one of the two ends in this case should be in I_t and it can alway be adjusted to make the constraint active.

   In short, we only need to find an edge whose ends are not in I_t. We then increase u_j until the first dual constraint gets active. There are only two constraints including u_j: the ones related to the two ends of the link. So basically, we choose the end v with smaller slack value $\epsilon = 1 - \sum_{j'|v \in e'_j} u_{j'}$ and add the value ϵ to u_j.

4. If there is no j to select stop. Otherwise, go to 2.

5. The primal solution $\mathbf{x}$, where $x_i = 1$ if an only $x_i \in I_t$ is our approximate solution.

We can clear up the above explanations and summarize the algorithm. Notice that the dual variables can only increase and the set I_t can only get larger. In fact, we do not need to calculate I_t at the second step, since bu calculating we have $I_{t+1} = I_t \cup \{v\}$, where v is the end node of the j^{th} link with smaller slack value than the other end, calculated in the step 3. We can explain these steps as below:

1. Start from $\mathbf{y}_0 = \mathbf{0}$ and $I_0 = \{\}$. Set $t = 0$.

2. Find a link e_j, which is not covered by I_t. If it does not exist, stop and return I_t as the set of nodes.

3. Calculate the slack values $\epsilon = 1 - \sum_{j'|v \in e'_j} u_{j'}$ for v, being the two ends of e_j. Select the smaller one. Call this node and its corresponding slack value v and ϵ, respectively.

4. Update $I_{t+1} = I_t \cup \{v\}$ and also u_j, corresponding to e_j, to $u_j + \epsilon$.

5. Update $t = t + 1$ and go to step 2.

Since the algorithm adds one node at each iteration, it will stop in at most n iterations, which impressively fast. How good the result is? It can be seen that the size of I_t at the stopping point cannot be larger than twice the size of the smallest covering set. Hence, the solution of the primal-dual method is a $2-$approximation of the minimum covering set problem. I will show this below, but first note that this result is promising. It is shown that if you use other "more intuitive" techniques the result can be 1000 times larger than the size of the minimum covering set. You might suspect that the number 1000 was arbitrary; You can end up in situation with k times larger result for any value of k. In many practical situations the solution of the primal-dual method is much better than twice the minimum size, although we currently do not have a good theory to explain it.

Now, let us prove that the primal-dual algorithm is a 2-approximation. We use the weak duality result. At the stopping point, the dual parameters $u_1, u_2, \ldots, u_m$ are feasible and so the dual cost $\sum_{k=1}^{m} u_m$ is a lower bound for the primal optimal value, i.e. the size $|I_{opt}|$ of the minimum covering set I_{opt}, i.e.

$$\sum_{k=1}^{m} u_m \leq |I_{opt}| \tag{69}$$

Then, the idea is to connect the size of the output $|I_t|$ to the dual parameters. Recall that for any $i \in I_t$, we have that its corresponding dual constraint is active, i.e. $\sum_{j|v \in e_j} u_j = 1$. Now, if I denote

$I_t = \{i_1, i_2, ..., i_q\}$, I can write

$$p = |I_t| = \underbrace{\sum_{j|v_{i_1} \in e_j} u_j}_{=1} + \underbrace{\sum_{j|v_{i_2} \in e_j} u_j}_{=1} + \ldots + \underbrace{\sum_{j|v_{i_p} \in e_j} u_j}_{=1} \tag{70}$$

Let us see how many times a particular dual variable u_j appears in the right hand side of the above. It is the number of nodes in I_t, which are connected to the j^{th} link. I show this by α_j. It should be clear that $0 \leq \alpha_j \leq 2$ as each link can be connected to at most two computers. Then, I can write (70) as

$$p = |I_t| = \alpha_1 u_1 + \alpha_2 u_2 + \ldots + \alpha_m u_m \leq 2 \sum_{i=1}^{m} u_m \tag{71}$$

Finally, (69) gives that $|I_t| \leq 2|I_{opt}|$.

Our primal-dual algorithm is not always guaranteed to give us the best possible result, but can be modified and improved. Often, the idea is that we should specify a "good" primal solution before taking a new variable. A good example is the assignment problem and the **Hungarian method**. In the next example we will see why our approach fails for the assignment problem, but will not explain the remedy (i.e., the Hungarian algorithm) in details.

Example 22. The assignment problem: In a company, a group of n applicants are selected to take n job positions. Each job is taken by exactly one applicant and each applicant takes exactly one job. However, there is a cost $c_{i,j}$ for assigning the job position i to the applicant j. The aim is to find an assignment with a minimal cost. Here is an example:

$$(c_{i,j}) = \begin{pmatrix} 1 & 2 & 3 & 3 \\ 3 & 4 & 4 & 2 \\ 1 & 2 & 4 & 3 \\ 3 & 1 & 2 & 4 \end{pmatrix} \tag{72}$$

To solve this problem, we take $x_{i,j} \in \{0, 1\}$ as the indicator that the job i is assigned to the applicant j. We can write the optimization as

$$\begin{gathered} \min_{(x_{i,j} \in \{0,1\})} \sum_{ij} c_{i,j} x_{i,j} \\ \text{subject to} \\ \forall i, \quad \sum_j x_{i,j} = 1 \\ \forall j, \quad \sum_i x_{i,j} = 1 \end{gathered} \tag{73}$$

The dual can be written as

$$\begin{gathered} \max_{(y_i, z_j)} \sum_i y_i + \sum_j z_j \\ \text{subject to} \\ \forall i, j \quad y_i + z_j \leq 1 \end{gathered} \tag{74}$$

It is possible to show that the underlying constraint matrix for this problem is totally unimodular. Hence, both the primal LP relaxation and dual problems have integer solutions. Here, we want to derive a primal-dual algorithm for this problem.

1. We start from a dual feasible solution. Take $y_i = 0$ and $z_j = 0$ in this case.
2. Similar to the previous example, we keep track of the set of active dual constraints $I = \{(i, j) \mid y_i + z_j = 1\}$. It is initiated by $I = \emptyset$.

3. At each iteration, we look for one of the dual variables y_i or z_j, such that its corresponding constraint is violated. Without loss of generality, suppose that the variable is y_i and $\sum_j x_{i,j} \leq 1$ for any choice of $x_{i,j}$ for $(i,j) \in I$. This means that for this value of i, no (i,j) is in I.

4. Choose ϵ as the minimum value of $1 - y_i - z_j$ over all values of z_j (or y_i if z_j is selected in the previous step) and update y_i to $y_i + \epsilon$.

5. stop when there is no more choices.

If we try the above scheme on the example in (72) and take the variables y_1, y_2, y_3, y_4, z_3, respectively (i.e, all rows in order and then the third column) the algorithm stops, with the dual value 6, but there is no primal feasible point satisfying the complementary slackness conditions and the dual can also be increased to 7 by the choice: $y_i = 0, 1, 0, -1$ and $z_i = 1, 2, 3, 1$.