6 The Integer Linear Programs

Integer Linear Programming (ILP) is one of the broadest areas of integer programming. Although one can easily conceive a "non-linear" integer program, still the term "integer programming" is sometimes exclusively used for ILPs. Nowadays, Mixed ILP (MILP) problems are also very popular in industry: You might want to switch on or off some machines (the integer part), while at the same time tuning the angle of some robot arms (the continuous part). The focus of this course is on ILPs and their algorithms. It should be readily seen that LPs are very much connected to ILPs. As we see in this chapter, the LP theory is indeed very helpful to devise ILP algorithms.

An ILP is an optimization problem, which is identical to a LP from every respect, except that its variable space is the set of all integers. If the constraints of an ILP restrict its variables to be either 0 or 1, it is called a **0-1 ILP**. The ILP problems are often expressed in the standard form:

$$\begin{aligned} &\min_{\mathbf{x}\in\mathbb{Z}^n} \mathbf{c}^T\mathbf{x} \\ &\mathbf{A}\mathbf{x} \leq \mathbf{b} \\ &\mathbf{x} \geq \mathbf{0} \end{aligned} \tag{59}$$

The equality constraints are not popular in ILPs. The reason is that they can easily get infeasible or dramatically change the feasible set.

Example 16. Consider two integer variables x, y and the constraint $x + y = 1$. One can easily write $x = 1 - y$ with an arbitrary integer y. Now, let us assume that there is a small change in the second coefficient and the constraint is changed to $x + 1.01y = 1$. Then, $x = 1 - 1.01y$ is not a solution for x as it might not be integer. In fact, in the new case the solutions for x and y are $x = 1 - 101k$ and $y = 100k$ for an arbitrary integer k. You see that even a small perturbation can entirely change the solution set. Now, consider $1.01x + 1.01y = 1$. This equation does not have any integer solution!

From the above example, it is seen that the integer linear equality constraints cannot be easily eliminated by solving the system of linear equations. There is another standard way to exchange any equality constraint with two inequalities: You can write $h(\mathbf{x}) = 0$ as $h(\mathbf{x}) \geq 0$ and $h(\mathbf{x}) \leq 0$. However, as I have already mentioned, it is unlikely to encounter equality constraints in a practical ILP problem. Every ILP problem can also be brought into the augmented form by introducing the slack variables. However, you should be careful that the slack variables are still continuous. Hence the augmented form of an ILP is in fact MILP.

Unlike LPs, there is no easy way to solve a generic ILP. In fact, the worst ILP problems are among the most difficult problems in the interesting family of non-deterministic polynomial (NP) problems[2]. Hence, there are a number of ideas that one can apply to an ILP and hope that at least the result is a good approximation algorithm.

A very simple, but still very popular way to treat an ILP is to expand the variable space from $\mathbb{Z}^n$ to $\mathbb{R}^n$ and keep the other parts unchanged. This is called **LP relaxation**, as the resulting optimization is LP. We have already encountered LP relaxation in Examples 12 and 13. Generally speaking, the LP relaxed optimization has lower optimal value than its corresponding ILP. We saw this fact in Examples 12 and 13 by strong duality and observing that they have identical dual optimizations. There is a much simpler way to see this: Indeed $\mathbb{R}^n$ is "larger" than $\mathbb{Z}^n$. Hence, any feasible solution of ILP is also a feasible solution of the relaxed ILP. This means that the relaxed LP can do "at least as good as" the ILP, simply by selecting the optimal solution of the ILP.

How about the optimal solutions? In general, the optimal solution of the LP relaxation is not necessarily integral (i.e. it might contain non-integer values). Unfortunately, no strong relation

[2]If you are familiar with the theory of computational complexity, you can understand it in a more technical way: The ILP is NP-complete, since other NP-complete problems such as the boolean satisfiability (SAT) problems can be reduced to ILPs.

between, a non-integral solution and the ILP solution is known. After obtaining the non-integral LP solution, one generally transforms this solution to an integral one by heuristic methods, which are generally known as **rounding**. For example, one might round the LP solution to the nearest integral point. However, this does not always lead to a good result.

Example 17. Take Example 13 again. The ILP solution is at $(5, 0)$, while rounding the LP solution $(1.95, 4.92)$ to the nearest integral point leads to $(2, 5)$, which is in no sense close to $(5, 0)$. Moreover, the point $(2, 5)$ does not satisfy the first constraint. So, it is not a feasible solution at all! We might instead round the LP solution to $(2, 4)$, which is feasible. The cost at this point is -4.56, while the ILP optimal cost is -5.

There are more sophisticated rounding schemes, which I am not going to talk about here. The LP solution can also be used as an input to a more sophisticated optimization technique. Another way to deal with the problem of non-integral solutions is to modify the LP solvers, such that they only search over integral feasible solutions. This is what we are going to talk about, but before that let me mention some further aspects of LP relaxation.

Although in many cases the solution of the relaxed LP is not integral, in some cases it can be. There are certain cases, where we can even guarantee that the solution is integral. The augmented form with a **totally unimodular matrix** is an important example.

Example 18. Consider the optimization in (59) and its LP relaxation:

$$\begin{aligned} \min_{\mathbf{x}\in\mathbb{R}^n} \ & \mathbf{c}^T\mathbf{x} \\ & \mathbf{A}\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq \mathbf{0} \end{aligned} \tag{60}$$

Assume that all the entries of $\mathbf{A}$, $\mathbf{b}$ and $\mathbf{c}$ are integer. Remember that the dual of (60) is

$$\begin{aligned} \min_{\mathbf{u}\in\mathbb{R}^n} \ & \mathbf{b}^T\mathbf{u} \\ & \mathbf{A}^T\mathbf{u} \leq \mathbf{c} \\ & \mathbf{u} \leq \mathbf{0} \end{aligned} \tag{61}$$

Suppose that $\mathbf{u}^*$ is a dual optimal solution. We want to find the primal solution from the complementary slackness conditions. The complementary slackness conditions give that

1. For any i with $u_i < 0$, we must have that the i^{th} row in $\mathbf{A}\mathbf{x} \leq \mathbf{b}$ holds with equality at the primal optimal solution.
2. For any index j with inactive dual constraint, we must have that $x_j = 0$ at the primal optimal solution.

Obtain a submatrix $\mathbf{S}$ of $\mathbf{A}$ by the following procedure: only keep those rows i of $\mathbf{A}$, which are mentioned in Item 1 above and discard all columns j of $\mathbf{A}$ mentioned in Item 2. Denote by $\mathbf{x}_c$ the nonzero component of $\mathbf{x}$ (See figure 5) i.e. the ones that are not set to zero by the Item 2 above. Also denote by $\mathbf{c}_c$, the part of vector $\mathbf{c}$ corresponding to the active constraints, the ones in Item 1. Now, for a proper choice of the dual solution, we should have that the unidentified part $\mathbf{x}_c$ of vector $\mathbf{x}$ is calculated by

$$\mathbf{S}\mathbf{x}_c = \mathbf{c}_c \tag{62}$$

From linear algebra, we know that the solution of this system of equations is rational (fractional), where the denominator is the determinant $\det(\mathbf{S})$ of $\mathbf{S}$. Now, we can conclude that the solution $\mathbf{x}_c$ is integral if $\det(\mathbf{S}) = \pm 1$. The problem is that when the dual solution is not given, we do not know $\mathbf{S}$. There is a special case, where we can conclude that the solution is integral, even if we do not know $\mathbf{S}$: When every invertible submatrix of $\mathbf{A}$ has determinant ± 1. This matrix is called **totally unimodular**. We conclude that the LP relaxation in (60) has at least one integral solution if the matrix $\mathbf{A}$ is totally unimodular. Notice that there still might exist non-integral solutions. It might also happen that a matrix $\mathbf{A}$ is not totally unimodular, but still an integer solution exists.

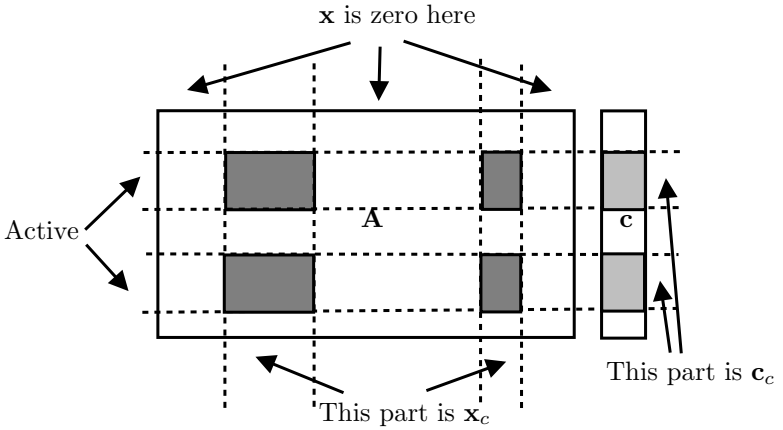

Figure 5: An example of $\mathbf{S}$ matrix and $\mathbf{x}_c$. The darker gray region is $\mathbf{S}$ and the lighter one is $\mathbf{c}_c$.

As soon as the solution of the LP relaxation is integral, it is also the solution of the ILP. The reason is simple: The LP searches over all its feasible solutions, including all integral feasible solutions, so its optimal integral point is certainly better than all other feasible integral points.

Example 19. In the minimum cost network flow problem in Example 15, suppose that we have an additional constraint that the flow has to be integral (has to have integer parameters). Let us call this the **minimum cost integer network flow** problem. For the LP relaxation, the underlying matrix is totally unimodular. Hence, as long as the costs c_k, balance terms b_k and the capacities u_k are integer, the LP relaxation has an integer solution. Hence, there is no integrality gap.

Example 20. From the definition, it is clear that when $\mathbf{A}$ is totally unimodular, so is its transpose $\mathbf{A}^T$. This means that when $\mathbf{A}$ is totally unimodular and all parameters are integer the dual optimization in (61) also has an integer solution. Remember that the dual of the minimum cost network flow problem is given in (55). For the special case of maximum s-t flow. We arrived at the optimization in (56). Since the corresponding matrix to the minimum cost network flow problem is totally unimodular, its dual also has an integer solution. This shows that even if we assume that the variables in (55) or more specifically (56) are integer, the optimal value and its integer solutions do not change. We call the resulting optimization the integer dual. Here is a summary: the primal ILP and its LP relaxation have the same optimal value and similar integer solutions. Also, the dual and its integer restriction have the same optimal value and similar integer solutions. This yields to two results: First, from strong duality for LPs, the minimum cost integer network flow and its *integer* dual have the same optimal value. Second, the complementary slackness conditions hold for all solutions of the minimum cost integer network flow and its integer dual.

Now, let us specialize these results for the maximum s-t flow problem. The integer dual is given by

$$\begin{gathered}\max_{(y_v\in\mathbb{Z}),(z_a\in\mathbb{Z})} \sum_{a\in A} u_a z_a \\ \forall a \in A\backslash\{a_0\},\ y_{t(a)} - y_{h(a)} + z_a \leq 0 \\ a = a_0 \Rightarrow y_{t(a)} - y_{h(a)} \leq -1 \\ \forall a \in A,\ z_a \leq 0\end{gathered} \tag{63}$$

First, notice that we can add any integer value n to all variables y_v and the cost will not change. This means that, I can make one variable, say y_{v_t} zero without decreasing the cost. Now, I show that the integer dual optimization above has a solution where $y_v \in \{0, 1\}$. In fact, I show

that there is a solution where over all y_v values, y_{v_s} is the maximal value, y_{v_t} is the minimal value and $y_{v_s} - y_{v_t} = 1$. In this case, I can reduce $n = y_{v_t}$ from every y_v value, which sets $y_{v_t} = 0$, $y_{v_s} = 1$ and all others to be either zero or one.

Now, let me show that there is a solution where over all y_v values, y_{v_s} is the maximal value, y_{v_t} is the minimal value and $y_{v_s} - y_{v_t} = 1$. Take any other optimal solution $\{y_v\}$ and $\{z_a\}$. If y_{v_s} is not the maximum value of y_v or $y_{v_s} - y_{v_t} > 1$, find all nodes with maximal y_v and reduce their value y_v by one unit. If y_{v_t} is not the minimum value, find all the nodes with the minimal value y_v and increase their value y_v with one unit. It remains as an exercise (not mandatory) to show that these steps keep the solution feasible. It is clear that the cost does not change as we don not change $\{z_a\}$ values. I can repeat this procedure until my conditions are satisfied. Finally, I arrive at my desired solution $y_v \in \{0, 1\}$ by changing all y_v values to $y_v - y_{v_t}$. Notice that each z_a satisfies two constraints $z_a \leq 0$ and $z_a \leq y_{h(a)} - y_{t(a)}$. So, the maximization will automatically activate one of these two. Notice that $y_{h(a)} - y_{t(a)}$ can only be $0, 1$ or -1. So, at the optimal solution $z_a \in \{0, -1\}$. It is now clear that $z_a = -1$ only happens when $y_{h(a)} = 0$ and $y_{t(a)} = 1$.

The fact that the integer dual optimization has a 0-1 solution means that we can restrict it more, without changing its optimal value or its 0-1 solutions:

$$\begin{gathered}\max_{(y_v \in \{0,1\}), (z_a \in \{0,-1\})} \sum_{a \in A} u_a z_a \\ \forall a \in A \backslash \{a_0\},\ y_{t(a)} - y_{h(a)} + z_a \leq 0 \\ y_{v_t} = 0 \\ y_{v_s} = 1\end{gathered} \tag{64}$$

Now, we can give an important interpretation about the optimization in (64). Define $S = \{v \in V \mid y_v = 1\}$. Then the optimization in (64) gives you the subset S of nodes such that the total cost of its corresponding sum,

$$\sum_{a | t(a) \in S,\ h(a) \in S^c} u_a \tag{65}$$

is minimized. The argument above shows that the minimum cost for the cut in the dual is equal to the maximum flow in the network. this is the min-cut-max-flow theorem.