4 Integer Linear Programming (ILP)

An integer linear program, ILP for short, has the same form as a linear program (LP). The only difference is that the feasible solutions are restricted to be integer vectors, i.e. $\mathbf{x} \in \mathbb{Z}^n$ instead of $\mathbf{x} \in \mathbb{R}^n$. All other elements such as $\mathbf{A}$ can still have real numbers as components. While it might seem intuitively like this should be a much easier problem (after all, we don't have infinitely many feasible values anymore), solving ILP is in fact *much harder*[1]! Part of the reason is that the feasible set is not convex anymore, since it consists of finitely many disjunct points.

If an LP has optimal solutions, then there will be an optimal solution which corresponds to one of the vertices in the feasible polytope. However, since the constraints in an ILP can be exactly the same as in the LP, there is no guarantee that this vertex will be integer. Therefore, obviously, $f^*_{\text{ILP}} \leq f^*_{\text{LP}}$ for a maximization problem, and $f^*_{\text{ILP}} \geq f^*_{\text{LP}}$ for a minimization problem. This also implies that in ILP the duality gap (the difference between a primal and a dual optimal solution) is not zero in general. Since an approximate solution is better than no solution at all, sometimes one solves the LP form of an ILP and then rounds to the closest feasible integer solution; this kind of approximation is called *LP relaxation*. However, that approximation can be arbitrarily bad, and it is easy to construct such cases (Can you find a way to do this?).

4.1 ILP with strong duality

Before we start dealing with general ILP, we look at an important special case: if the constraints in an ILP are such that they intersect at an integer coordinate for the optimal value, we can actually use the LP relaxation to solve the ILP optimally! Furthermore, the duality gap is zero in that case, and the problem exhibits *strong integer duality*.

The question then arises how we can know whether or not a given ILP can be solved optimally by its LP relaxation. In theory, only the optimizing vertex would have to be integer. However, when we want to decide whether an LP relaxation will yield an optimal ILP solution, we cannot know which vertex corresponds to the optimal solution without solving the program explicitly. Therefor, it would be nice to guarantee that *all* vertices are integer, and hence the program works for arbitrary objectives. An example

[1] In the language of complexity theory, LP is in P, while ILP is NP-hard! We'll get to the details in a later lecture

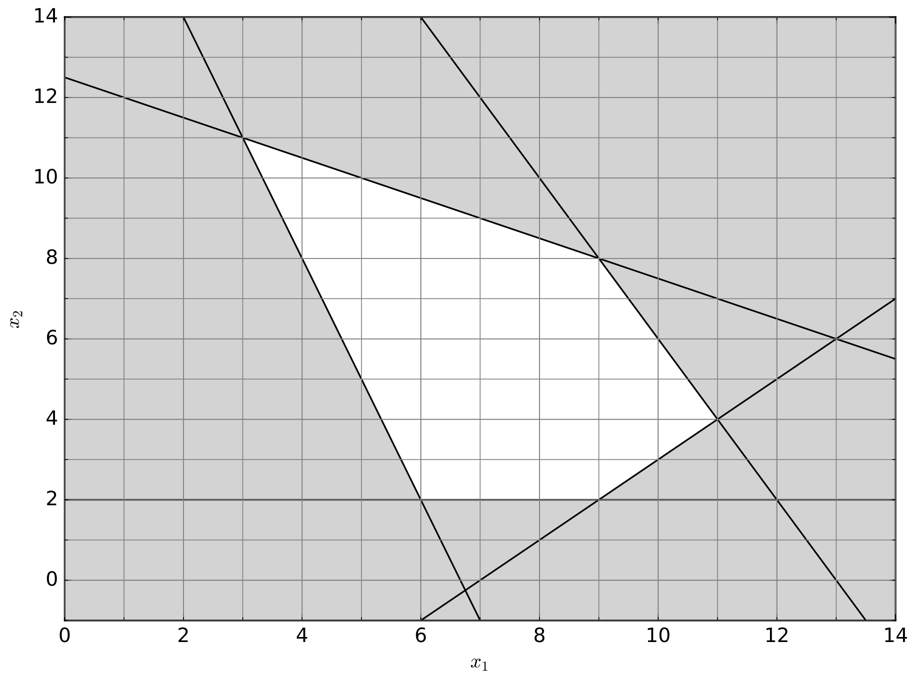

Figure 1: A feasible polytope with integer vertices (this is *not* a TUM). In cases like this, the LP relaxation solves the ILP optimally for any objective function.

of such a polytope is given in fig. 1. It is described by the inequality system

$$\begin{pmatrix} 1 & 2 \\ 2 & 1 \\ 1 & -1 \\ 0 & -1 \\ -3 & -1 \end{pmatrix} \mathbf{x} \leq \begin{pmatrix} 25 \\ 26 \\ 7 \\ -2 \\ -20 \end{pmatrix}$$

In $\mathbb{R}^n$, the location of a vertex is the intersection of n hyperplanes which correspond to the boundaries of halfspaces defined in the constraints. In other words, each vertex is the unique solution of some system of n linear equations (e.g. in $\mathbb{R}^2$, we need to intersect two lines to get a single point, in $\mathbb{R}^3$, we need to intersect three non-parallel planes to get a point, but two planes only get us a line). So there is some invertible matrix $\mathbf{S}$ such that $\mathbf{Sx} = \mathbf{b}$ and $\mathbf{x}$ has an integer solution (it has to be invertible because otherwise we could not solve $\mathbf{x} = \mathbf{S}^{-1}\mathbf{b}$). Whether or not this holds depends on $\mathbf{b}$; indeed, if we changed $\mathbf{b}$ in fig. 1, the lines would move and their intersections might become non-integer. So for general problems, our safest bet is a matrix which has integer solutions $\mathbf{x}$ for all integer $\mathbf{b}$. Such a matrix is called *unimodular*.

Definition 1 (Unimodular matrix)**.** *An integer square matrix* $\mathbf{S} \in \mathbb{Z}^{n\times n}$ *is called a* unimodular matrix (UM) *iff* $\det \mathbf{S} = \pm 1$.

Note that this implies that $\mathbf{S}$ is invertible (non-singular, i.e. $\mathbf{S}^{-1}$ exists), as $\det \mathbf{S} \neq 0$.

Theorem 1. *Let* $\mathbf{S} \in \mathbb{Z}^{n\times n}$. *Consider the system of linear equations* $\mathbf{S}\mathbf{x} = \mathbf{b}$. *Then* $\mathbf{x}$ *is integer* ($\mathbf{x} \in \mathbb{Z}^n$) *for any* $\mathbf{b} \in \mathbb{Z}^n$ *iff* $\mathbf{S}$ *is UM.*

Proof. By Cramer's rule,
$$x_i = \frac{\det \mathbf{S}_i}{\det \mathbf{S}}$$
where $\mathbf{S}_i$ is $\mathbf{S}$ with the i-th column replaced by $\mathbf{b}$, which for general $\mathbf{b} \in \mathbb{Z}^n$ holds only if $\det \mathbf{S} = \pm 1$. Conversely, let $\mathbf{b}_t$ be a vector of zeros, with 1 at its t-th position. Since $\mathbf{x} = \mathbf{S}^{-1}\mathbf{b}$ for all integer $\mathbf{b}$, we must have $\mathbf{x} = \mathbf{S}^{-1}\mathbf{b}_t$ for all t, in which case the t-th column of $\mathbf{S}^{-1}$ is $\mathbf{x}$, and hence integer. Hence all columns of $\mathbf{S}^{-1}$ are integer, and therefore $\det \mathbf{S}^{-1}$ is integer. Since $\det \mathbf{S}^{-1} = (\det \mathbf{S})^{-1}$, $(\det \mathbf{S})(\det \mathbf{S}^{-1}) = 1$, and therefore $\det \mathbf{S} = \det \mathbf{S}^{-1} = \pm 1$. So $\mathbf{S}$ must be UM. □

This gets us almost to where we want to be. Consider the primal constraints
$$\mathbf{A}\mathbf{x} \le \mathbf{b}$$
$$\mathbf{x} \ge \mathbf{0}$$
We can rewrite this as a concatenation of two matrices,
$$\begin{bmatrix} \mathbf{A} \\ -\mathbf{E} \end{bmatrix} \mathbf{x} \le \begin{bmatrix} \mathbf{b} \\ \mathbf{0} \end{bmatrix}$$
where $\mathbf{E}$ is the identity matrix. This neatly describes the feasible polytope using a single matrix. Now, obviously, each vertex is described by a system of linearly independent linear equations $\mathbf{A}^\circ\mathbf{x} = \mathbf{b}^\circ$, where[2]
$$\mathbf{A}^\circ \sqsubseteq \begin{bmatrix} \mathbf{A} \\ -\mathbf{E} \end{bmatrix}$$
and the corresponding entries
$$\mathbf{b}^\circ \sqsubseteq \begin{bmatrix} \mathbf{b} \\ \mathbf{0} \end{bmatrix}$$

[2]We use $\sqsubseteq$ to denote submatrices and subvectors.

As each vertex is supposed to be integer, each $\mathbf{A}^\circ$ must be unimodular. There is a problem though: some intersections are not parts of the polytope, and would not necessarily have to be UM. Consider the intersection in the vicinity of $(7, 0)$ in fig. 1 (Which constraints define this intersection?). It is not integer, which is no problem as it is not a vertex in the polytope. However, if we dropped the constraint $-x_2 \le -2$, it would be part of the polytope, and yield a non-integer solution for certain objectives. Identifying which intersection points are polytope vertices and thus which submatrices have to be UM is not straightforward. Therefor, our safest bet is to simply require all invertible submatrices to be unimodular. This leads to the following:

Definition 2 (Totally unimodular matrix)**.** *A matrix* $\mathbf{A}$ *is called a* totally unimodular matrix (TUM) *iff every invertible submatrix* $\mathbf{S} \sqsubseteq \mathbf{A}$ *is unimodular. Equivalently,* $\mathbf{A}$ *is a TUM iff for all its square submatrices* $\mathbf{S}$, $\det \mathbf{S} \in \{-1, 0, 1\}$.

This implies that all entries are $-1, 0$ or 1, since all submatrices of size 1×1 have to be UM as well, and $\det(a) = a$. Note that this is not a sufficient condition, not all matrices which only have entries like that are TUM.

So, whatever square submatrix we pick, it is either singular (determinant is 0), so it cannot define a vertex, or it is invertible and defines a vertex, and we know that the vertex is integer, since the submatrix is guaranteed to be unimodular. So ILP constraints defined by a matrix $\mathbf{A}$ which is TUM guarantee that all vertices are integer, hence an optimal solution to the LP relaxation is integer, and we can easily solve the ILP using the LP. It should be noted that being TUM is a *sufficient*, but not a *necessary* condition for integrality. Clearly, the polytope in fig. 1 is not a TUM, as its matrix contains many entries that are not 0 or ± 1, yet it only has integer vertices. TUM is a rather restrictive condition, yet it occurs in practice, especially in optimization problems involving *directed graphs* (or *digraphs* for short).

Before we look at some examples, I'd like to make duality a little more explicit. In the last lectures we've looked at primal-dual pairs in standard form, and how we can change constraints of a general LP to make it fit that form. However, we don't always want to do this, and it will be useful to talk about duality in a more general way. There is an easy way to express duality for LPs in arbitrary form (fig. 2). Notice for instance that the dual variable of an equality constraint is unconstrained (here: y_2 and c_2). For a $\ge$-constraint in a maximization problem, the dual variable is non-positive (y_3). For inequality constraints that point into the opposite direction than what we would expect from the standard form, the bounds of the dual variables also change direction (here: $x_3 \le 0$, because c_3 is bounded from below instead of above, and $y_3 \ge 0$, because b_3 is bounded from above instead of below). Note that this general duality contains our standard form as a special case. Some authors also consider the special case in which all primal (dual) variables are unconstrained. These pairs are sometimes called

$$
\begin{array}{|c|ccc|ccc|cc|}
\hline
 & & & & \multicolumn{3}{c|}{\max \mathbf{c}^\intercal \mathbf{x}} & & \\
\hline
 & & & & 0 & & 0 & & \\
 & & & & \wedge\!| & & | \!\vee & & \\
 & & & & x_1 & x_2 & x_3 & & \\
\hline
 & 0 & \leq & y_1 & a_{11} & a_{12} & a_{13} & \leq & b_1 \\
\min \mathbf{b}^\intercal \mathbf{y} & & & y_2 & a_{21} & a_{22} & a_{23} & = & b_2 \\
 & 0 & \geq & y_3 & a_{31} & a_{32} & a_{33} & \geq & b_3 \\
\hline
 & & & & | \!\vee & \| & \wedge\!| & & \\
 & & & & c_1 & c_2 & c_3 & & \\
\hline
\end{array}
$$

Figure 2

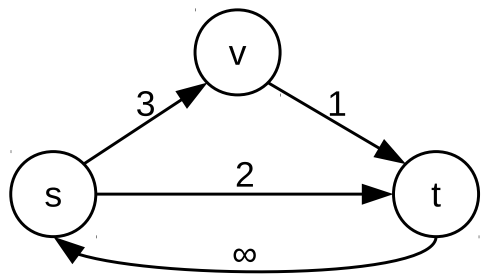

Figure 3

asymmetric duals:

$$
\begin{array}{ll@{\qquad}ll}
\max & \mathbf{c}^\intercal \mathbf{x} & \min & \mathbf{b}^\intercal \mathbf{y} \\
\text{s.t.} & \mathbf{A}\mathbf{x} \leq \mathbf{b} & \text{s.t.} & \mathbf{A}^\intercal \mathbf{y} = \mathbf{c} \\
 & & & \mathbf{y} \geq \mathbf{0}
\end{array}
$$

and

$$
\begin{array}{ll@{\qquad}ll}
\max & \mathbf{c}^\intercal \mathbf{x} & \min & \mathbf{b}^\intercal \mathbf{y} \\
\text{s.t.} & \mathbf{A}\mathbf{x} = \mathbf{b} & \text{s.t.} & \mathbf{A}^\intercal \mathbf{y} \geq \mathbf{c} \\
 & \mathbf{x} \ \ \geq \mathbf{0} & &
\end{array}
$$

Now let's look at the maximum s-t flow problem. A very small instance[3] of this problem is shown in fig. 3. We have one source node s and one sink node t. Arcs (edges) are labeled using capacities. We send some flow from the source to the sink along the arcs. The flow x_{ij} along an arc (i, j) must not exceed its capacity (*capacity constraints*), and must not be negative (*non-negativity constraint*). The sum of flows into a node must equal the sum of flows out of a node (*flow balance constraint*, due to its applications in physics this is sometimes called *mass conservation constraint*). A virtual edge (t, s) of infinite capacity has been added to allow the flow to be balanced. Our objective is to maximize the flow from source to sink; due to flow balance, this is equivalent to maximizing the flow from t to s. This problem is important for several reasons. For integer capacities and flows, its LP relaxation is integer, and thus it can be solved efficiently, as can its dual. It also turns out that the dual solution only requires values in $\{0, 1\}$ instead of $\mathbb{Z}$, and is thus an instance of so-called *0-1 linear programming*, an important special case of ILP. The primal of our toy example has the form

[3]This example is due to Marc Uetz.

$$
\begin{array}{lr}
\max\limits_{\mathbf{x}\in\mathbb{Z}^m} & x_{ts} \\
\text{s.t.} & x_{sv} + x_{st} - x_{ts} = 0 \\
 & -x_{sv} + x_{vt} = 0 \\
 & -x_{st} - x_{vt} + x_{ts} = 0 \\
 & x_{sv} \leq 3 \\
 & x_{st} \leq 2 \\
 & x_{vt} \leq 1 \\
 & x_{ts} \leq \infty \\
 & \mathbf{x} \ \ \geq 0
\end{array}
$$

where the first three constraints are the mass balance, the next four the capacity and the last one the non-negativity constraint. Using the scheme for the dual in general form, we can write this as in fig. 4. Notice that the matrix is a concatenation $\begin{bmatrix}\mathbf{A}\\ \mathbf{E}\end{bmatrix}$ of the incidence matrix $\mathbf{A}$ of the network and the identity matrix $\mathbf{E}$. The incidence matrix of a *directed graph* of n nodes and m arcs is an $\{-1, 0, 1\}^{n\times m}$ matrix, where $a_{v,r}$ is 1 if arc r points away from node v, -1 if it points towards v, and 0 otherwise. It turns out that this matrix, as well as its concatenation with $\mathbf{E}$ is TUM and therefore observes

$$\max \begin{pmatrix}0\\0\\0\\1\end{pmatrix}^{\intercal}\begin{pmatrix}x_{sv}\\x_{st}\\x_{vt}\\x_{ts}\end{pmatrix}$$

$$\min \begin{pmatrix}0\\0\\0\\3\\2\\1\\\infty\end{pmatrix}^{\intercal}\begin{pmatrix}\pi_s\\\pi_v\\\pi_t\\\alpha_{sv}\\\alpha_{st}\\\alpha_{vt}\\\alpha_{ts}\end{pmatrix}$$

$$\begin{array}{ccc|cccc|cc}
 & & & 0 & 0 & 0 & 0 & & \\
 & & & \leq & \leq & \leq & \leq & & \\
 & & & x_{sv} & x_{st} & x_{vt} & x_{ts} & & \\
\hline
 & & \pi_s & 1 & 1 & 0 & -1 & = & 0 \\
 & & \pi_v & -1 & 0 & 1 & 0 & = & 0 \\
 & & \pi_t & 0 & -1 & -1 & 1 & = & 0 \\
0 & \leq & \alpha_{sv} & 1 & 0 & 0 & 0 & \leq & 3 \\
0 & \leq & \alpha_{st} & 0 & 1 & 0 & 0 & \leq & 2 \\
0 & \leq & \alpha_{vt} & 0 & 0 & 1 & 0 & \leq & 1 \\
0 & \leq & \alpha_{ts} & 0 & 0 & 0 & 1 & \leq & \infty \\
\hline
 & & & \geq & \geq & \geq & \geq & & \\
 & & & 0 & 0 & 0 & 1 & &
\end{array}$$

Figure 4

strong integer duality! The general form of this LP is

$$\begin{aligned}
\max\quad & \mathbf{c}^{\intercal}\mathbf{x} \\
\text{s.t.}\quad & \mathbf{Ax} = \mathbf{0} \\
& \mathbf{x} \;\leq \mathbf{b} \\
& \mathbf{x} \;\geq \mathbf{0}
\end{aligned}$$

From the representation above it can easily be seen that the dual is

$$\begin{aligned}
\min_{\boldsymbol{\alpha}\in\mathbb{Z}^n}\quad & 3\alpha_{sv} + 2\alpha_{st} + \alpha_{vt} + \infty\alpha_{ts} \\
\text{s.t.}\quad & \alpha_{sv} + \pi_s - \pi_v \geq 0 \\
& \alpha_{st} + \pi_s - \pi_t \geq 0 \\
& \alpha_{vt} + \pi_v - \pi_t \geq 0 \\
& \alpha_{ts} + \pi_t - \pi_s \geq 1 \\
& \boldsymbol{\alpha} \;\geq \mathbf{0}
\end{aligned}$$

Notice that, due to the equality constraints in the primal, $\boldsymbol{\pi}$ is unconstrained. The general form is

$$\begin{aligned}
\min\quad & \begin{bmatrix}\mathbf{0}\\\mathbf{b}\end{bmatrix}^{\intercal}\begin{bmatrix}\boldsymbol{\pi}\\\boldsymbol{\alpha}\end{bmatrix} \\
\text{s.t.}\quad & [\mathbf{A}^{\intercal},\mathbf{E}^{\intercal}]\begin{bmatrix}\boldsymbol{\pi}\\\boldsymbol{\alpha}\end{bmatrix} \geq \mathbf{c} \\
& \boldsymbol{\alpha} \qquad\qquad \geq \mathbf{0}
\end{aligned}$$

It is clear from the indices that α_{ij} is a dual variable associated with arc (i,j), and π_i is a dual variable associated with node i. In fact, each constraint relates three entities: an *arc variable* α_{ij}, a *head variable* π_j and a *tail variable* π_i. Except for the back arc (t,s) these constraints say that the arc variable has to be at least as large as the difference between the head and the tail variable ($\alpha_{ij} \geq \pi_j - \pi_i$). Now, the central observation is that we can add any constant to $\boldsymbol{\pi}$, because that constant cancels out in the dual constraints, and changes to $\boldsymbol{\pi}$ do not change the objective, since it does not appear there. Without loss of generality, we add a constant to $\boldsymbol{\pi}$ such that $\pi_s = 0$. Then, since the constraint $x_{ts} \leq \infty$ has slack for any finite valuation, we conclude $\alpha_{ts} = 0$ from complementary slackness, thus making the objective function finite. It follows that $\pi_t \geq 1 + \pi_s - \alpha_{ts} = 1$. To minimize, we'd like to set all α_{ij} to zero, but some constraints get in the way. Since $\pi_s = 0$ and $\pi_t \geq 1$, any path from s to t has to have at least one arc (i,j) for which the head variable π_j is strictly greater than the tail variable π_i (in all other cases, i.e. when $\pi_j \leq p_i$, we can set α_{ij} to zero). Consequently, since there exists an optimal integer solution, on each path there is an arc variable $\alpha_{ij} \geq 1$. It turns out that we can set $\pi_t = 1$, so any s-t-path will have an arc for which $\alpha_{ij} = 1$, and we can restrict ourselves to $\pi_i, \alpha_{ij} \in \{0,1\}$ instead of $\pi_i, \alpha_{ij} \in \mathbb{Z}$. Removing the set of all such arcs disconnects the network into two components, since it disconnects the source and sink for all paths. The first component contains s and all $\pi_i = 0$, the other contains t and all $\pi_i = 1$. Therefor, π_i are the node labels telling us which component a node belongs to, and the arc variables tell us which arcs we need to select to disconnect the network. For obvious reasons, such an arc set is called a *cut*, and since the coefficients in the minimizing objective function in the dual are the capacities of those edges, this is the *minimum capacity cut*, or *minimum cut* for short. By strong duality (due to TUM), the value of the dual objective function (minimum cut) is the same as the one for the primal (maximum flow). This surprising duality is better known as the *max-flow min-cut theorem*.