Lecture 7 - Integrality of polytopes

Optimization and Approximation - ENS M1

Nicolas Bousquet

1 Introduction

In previous lectures, we have seen that we have efficient algorithms to solve real Linear Program while such algorithms do not exist for Integer Linear Programs in general (actually they are usually not approximable). In this lectures, we will see several examples of polytopes of a particular type: all their extreme points are integral points. Since the Simplex algorithm provide an extreme point, it implies that the optimal solution of any maximization / minimization on these polytopes are integral.

A vector is *integral* if all its coordinates are integers. A polytope is *integral* if all its extreme points are integral vectors.

2 TU matrices

Let M be a real $m \times n$ matrix. A squared submatrix M' of M is a matrix of size $k \times k$ for some k such that there exists k rows R and k columns C such that M' is the matrix whose coefficients correspond to the coeffient of M that are in both the rows of L and the columns of C. (Note in particular that rows and columns are not necessarily consecutive, a rectangle is a "combinatorial" rectangle). A matrix M is *totally unimodular* (TU for short) if every squared submatrix of M has determinant in $\{0, +1, -1\}$. In particular every entry of M must be in $\{0, +1, -1\}$ because every entry is a trivial square submatrix.

Lemma 1. *Let M be a totally unimodular matrix. Let b and c be integer vectors. Then*

- *Every extreme point of $P = \{x : Mx \leq b\}$ is an integral vector.*
- *Every extreme point of $P = \{x : Mx = b, 0 \leq x \leq c\}$ is an integral vector.*

We know that extreme points are equivalent to basic feasible solutions so we prove lemma for basic feasible solutions:

Lemma 2. *Suppose A is totally unimodular matrix. Let b be any integer vector. Then every basic feasible solution of $P = \{x : Ax \leq b\}$ has integer coefficients.*

Proof. Let x be a basic feasible solution. Then the constraints that are tight at x have rank n (indeed a point is the intersection of n hyperplanes). Let A' be a submatrix of A and b' a sub-vector of b corresponding to n linearly independent constraints that are tight at x. Then x is the unique solution to $A'x = b'$, *i.e.* $x = (A')^{-1}b'$.

In Linear Algebra you (must) have seen that if M is a square, non-singular (*i.e.* which can be inverted) matrix then

$$(M^{-1})_{i,j} = (-1)^{i+j} \frac{det(M_{del(j,i)})}{det(M)}$$

In which $detM_{del(j,i)}$ is the submatrix of M obtained by deleting row j and column i. (in other words the inverse of a matrix equals the transpose of the matrix of cofactors divided by the determinant).

Thus all entries of $(A')^{-1}$ are in $\{0, +1, -1\}$. Since b' is an integer vector, x also is. □

We left as an exercise to show that the following holds:

Exercise 1. *Prove that*

- *Show that the transpose of a totally unimodular matrix is totally unimodular.*
- *Show that if M is totally unimodular then $\begin{pmatrix} M \\ I \end{pmatrix}$ is totally unimodular.*
- *Show that a submatrix of a totally unimodular matrix is unimodular.*

2.1 Example 1 - Incidence matrix of bipartite graph

Let $D = (V, E)$ be a graph. The *incidence matrix* M of G is the matrix of size $|E| \times |V|$ such that

$$M(e, u) = \begin{cases} 1 & \text{if } v \text{ is an endpoint of } e \\ 0 & \text{otherwise} \end{cases}$$

Exercise 2. *Prove that the incidence matrix of an odd cycle is not TU. Deduce that the incidence matrix of any non-bipartite graph is not TU.*

Let us prove that it is actually the only case where the matrix is not TU. Indeed, we have the following:

Theorem 3. *The incidence matrix of a bipartite graph is TU.*

Proof. Note that any row of the matrix has two non-zero entries (equal to 1). Let us prove it by induction on the size of the submatrix. It is indeed true for any 1×1 matrix since the coefficients are in $\{0, 1\}$. Now consider a submatrix M' of M of size $k \times k$. If there is one row with no non-zero entries, the determinant is equal to 0. If there is one row with exactly one non-zero entry, then we can develop the determinant for this row and we immediately obtain the conclusion by induction. So in what follows, we can assume that there are exactly two non-zero coefficient per row.

We claim that there is a combination of the columns that sums to 0 (which implies that the determinant is equal to 0). In order to prove it, let us partition the columns of M' according to their belonging to one side or the other of the bipartition. Indeed, by definition, each column is associated to a vertex. Now let C_1 be the subset of columns in the left part of the bipartite and C_2 be the subset of columns in the right part of the bipartition. By definition, every edge has one endpoint in C_1 and one endpoint in C_2. So on each row, there is one "1" in the set of rows corresponding to C_1 and one "1" in the set of columns corresponding to C_2. Now, for every C let y_C be the vector of M' corresponding to column C. We have:

$$\sum_{C \in C_1} y_C - \sum_{C \in C_2} y_C = 0$$

which ensures that the determinant is equal to 0. □

2.2 Example 2 - Incidence matrix of directed graphs

Let $D = (N, A)$ be an oriented graph. The *incidence matrix* M of D is the matrix of size $|A| \times N$ such that

$$M(a, j) = \begin{cases} +1 & \text{if } j \text{ is the beginning of the arc } a \\ -1 & \text{if } j \text{ is the ending of the arc } a \\ 0 & \text{otherwise} \end{cases}$$

(we say that j is respectively the head or the tail of a).

Theorem 4. *The incidence matrix of any directed graph is TU.*

In order to prove it, we just have to prove the following:

Exercise 3. *(i) What is the maximum number of non-zero coefficients on a row of the oriented matrix M of a graph D?*

(ii) Let Q be a squared submatrix of M. Show that if one row of Q has no non-zero entry, $det(M) \in \{-1, 0, +1\}$. Prove the same if one row has exactly one non-zero entry.

(iii) In the last case, prove that there is a combination of the columns that sums to 0...

3 Applications - Integrality gaps and polynomial time algorithms

3.1 Polytime algorithms

We have seen in the first lecture that the vertex cover problem can be formulated as follows:

$$\min \sum_{v \in V} x_v$$

subject to

$$x_u + x_v \geq 1 \text{ for all } e = (u, v) \in E$$
$$x_u \in \{0, 1\}$$

One can remark that the constraint matrix is precisely the incidence matrix of the graph. Thus even if the Minimum Vertex Cover problem is NP-complete in general graphs, it can be solved in polynomial time for bipartite graphs since the optimal value of the LP in real value also is the optimal value of the corresponding ILP.

3.2 Integrality gaps

Let (P) be an Integer Linear Program (ILP for short). The *fractional relaxation* of (P), denoted by (P*) is the LP where the Integrality constraint is replaced by a non-negativity constraint. In the particular case of $0-1$-Linear Program (ILP where variables can take either the value 0 or 1), the constraint $x \in \{0, 1\}$ can be replaced by the two constraints $0 \leq x \leq 1$.

One may ask what is the difference between the optimal value of the ILP and of its fractional relaxation. The *integrality gap* of (P) is $\frac{OPT(P')}{OPT(P)}$ if P is a maximization function and $OPT(P)/OPT(P')$ if (P) is a minimization function. Note that the integrality gap is always at least one since an integral solution always is a solution of its fractional relaxation. The question then is: how large can be the integrality gap? In general the integrality gap can be arbitrarily large, but when we can bound the integrality gap, we can often derive some interesting combinatorial and algorithmic properties.

Let (P) be an ILP and (D) be its dual (ILP). The *primal-dual gap* is the $\frac{OPT(D)}{OPT(P)}$.

Bounding the integrality gap is in general an interesting property. Indeed, if the integrality gap is bounded, we often have a "natural" road-map to find an approximation algorithm:

- Find an optimal solution of the fractional relaxation (in polynomial time).
- Round the values of the variables of the fractional relaxation to transform it into a solution of the original LP.

Even if this method does not necessarily work each time, a LP with an arbitrarily integrality gap has less chance to have an approximation algorithm (at least using LP).

Example 1: Vertex Cover - Matching in bipartite graphs One can easily prove that the dual problem of the Minimum Vertex Cover problem is the Maximum Matching problem. Since the objective vector and the vector of constraints are integral (and since the transpose of a TU matrix is TU), both the fractional relaxation of the matching problem and of the vertex cover problem admit integral solutions. So the (integral) optimal solutions of the primal and the dual are equal. It implies the following theorem, due to Konig:

Theorem 5 (Konig)**.** *In bipartite graphs, the maximum size of a matching is equal to the minimum size of a vertex cover.*

Note that it is not true in general. What can you say about the gap between these two values?

Example 2: Max flow - Min Cut Let us prove the classical Max-Flow Min-Cut theorem. (Formal definitions of all the notions of the theorem will be given a bit later).

Theorem 6 (MaxFlow-MinCut theorem (Ford and Fulkerson 1956))**.** *The minimum capacity of an $s-t$-cut is equal to the maximum value of an $s-t$-flow whenever all the capacities and the weights are integers.*

$s-t$-flows. We are given a *directed graph* (or a *network*) $G=(V,A)$, where $V=\{1,\dots,N,s,t\}$ is the set of *nodes* (or *vertices* but we will prefer node to avoid confusion) and A is the set of directed arcs (an arc is an ordered pair of vertices). For every arc uv, u is called the *beginning* (or the *origin* or the *tail* of the arc) while v is called the *end* (or the *head*) of the arc. The node s is called the *source* and the node t is called the *sink* of the graph. We assume that there are no arcs going into the source s, and there are no arcs coming out of the sink t. For every arc (i,j) (*i.e.* arc from i to j), we are given also a capacity $c_{ij}\geq 0$. For every arc ab, the node a is called the *origin* of the arc and the node b is called the *destination* of the node. This network could represent for example a sewage system, where the arcs represent water pipes, and the nodes represent connection points. Or the network could represent a computer network, where the arcs represent optical fiber cables and the nodes represent hubs. There are many other conceivable settings (road network, electricity network...).

Suppose that we want to send units of 'flow' (representing e.g. water or data packets) along the arcs (in the direction of the arc) from node to node, injecting flow into the network at s (the source) and taking flow out of the network at t (the sink). We assume that there is no leakage in the system, so that the amount of flow that goes into an arc equals the amount of flow that comes out of it and, at every node, the amount of flow that comes into the node equals the amount of flow that goes out of the node. This assumption is called the flow preservation assumption. Also, the amount of flow on arc (i,j) can be at most its capacity c_{ij}.

Formally a *$s-t$-flow* is a function $x: A\longrightarrow \mathbb{R}$ such that:

- $x(ij)\geq 0$.
- $x(ij)\leq c_{ij}$
- For every $i\neq s,t$, we have $\sum_{j/ji\in A} x(ij)=\sum_{j/ji\in A} x(ji)$ (Kirchhoff law).

Note that the last equation (Kirchhoff law) just asserts that the amount of flow "leaving" i is precisely the amount of flow "entering" in i. In electricity, Kirchhoff law ensures that the amount of intensity entering at each node is precisely the intensity leaving the node. This rule is also called the flow preservation constraint or energy preservation constraint. It essentially mean that the node (distinct from s and t) cannot create any flow but they can just forward the flow they receive.

Here, $x(i,j)$ represents the amount of flow going from i to j. Observe that we do not have flow preservation constraints for s and t. Actually our goal is to compute the maximum flow which can be "transported" from s to t. We call the total amount of flow going out of the source s the *value* of the $s-t$-flow. This amount is equal to the total amount of flow coming into the sink t (this follows from the flow preservation assumption). So the value of the $s-t$-flow is the total amount of flow that we send from s into t. Formally, the value of an $s-t$-flow x is:

$$\sum_{sj\in A} x(sj)=\sum_{jt\in A} x(jt)$$

Exercise 4. *Why do we have this second equality?*

A natural question can be raised: how much flow can we send through the network? In other words, what is the maximum value of an $s-t$-flow in G when taking into account the capacity constraints? This is called the MaxFlow problem:

The Max-Flow Problem:
Given a directed graph $G=(V,A)$, where $V=\{1,\dots,N,s,t\}$, a source s, a sink t, and capacities $c_{ij}\geq 0$, what is the maximum value of an $s-t$-flow in G?

We have seen that the Max-Flow problem can easily be expressed with a few constraints, and one can note that these constraints are linear. So the Max-Flow problem can be seen as a Linear Program:

$$z^* = \max \sum_{j/jt \in E} c_{jt} x_{jt}$$

under

$$\sum_{j/ji \in A} x_{ij} = \sum_{j/ji \in A} x_{ji} \text{ for every } i \neq s, t \quad (1)$$

$$x(ij) \leq c_{ij} \text{ for every } ij \in E$$

$$x_{ij} \geq 0 \text{ for every } ij \in E$$

$s-t$-cuts. Let us define another notion on graphs, called $s-t$-cuts. We will see that these two notions are related since the dual of a flow is a cut and the dual of a cut is a flow. A *$s-t$-cut* is a subset of arcs A' such that the deletion of A' separates s from t, *i.e.* the graph $G' = (V, A \setminus A')$ has no directed $s-t$-path. A *minimum $s-t$-cut* is an $s-t$-cut of minimum weight. The weight of the cut is often called the *capacity* of the cut.

Before studying further the link between these two notions, let us prove the following simple fact:

Lemma 7. *The maximum value of an $s-t$-flow is smaller than the minimum value of an $s-t$-cut.*

Proof. Assume that there is an $s-t$-cut Y of weight ω. Then the deletion of the edges of Y provides a graph where the maximum value of an $s-t$-flow is equal to 0 since there is no path from s to t anymore and then no way to transport flow from s to t.

Let $a_1, \ldots, a_k$ be k arcs of an s-t-cut. Let us show that the value of a $s-t$-flow is at most the sum of the weights of these arcs. Now consider the path representation of the $s-t$-flow problem. First note that all the paths from s to t pass through one of these arcs (otherwise it is not a $s-t$-cut by definition of $s-t$-cut). To every positive path in a solution of the flow problem, we can assign this path to an arc in $a_1, \ldots, a_k$ contained in the path.

The constraints of the path representations asserts that the sum of the flows of the paths passing through a_i is at most $w(a_i)$ for every i. So the total sum is at most $\sum w(a_i) = \omega$, which achieves the proof. □

Note that there is some similarities between this proof and the proof of the weak duality theorem... We'll see further that it is logical since it is the proof of the weak duality theorem for a particular LP...

The Min-Cut Problem:
Input: A directed graph $G = (V, A)$, where $V = \{1, \ldots, N, s, t\}$, s is a source and t a sink, and capacities $c_{ij} \geq 0$.
Output: The minimum capacity of an $s-t$-cut.

Let us give a LP formulation of the Min-Cut problem. Assume that the variables y_i and and y_{ij} are integers (*i.e.* $y_{ij} \in \{0, 1\}$ and $y_i \in \{0, 1\}$). We moreover consider that $y_s = 1$ and $y_t = 0$. A cut will be represented by a set of vertices containing s and not containing t. The size of the corresponding cut is the number of edges starting in X and ending in $V \setminus X$. So now if we can represent a LP where:

- Vertices receive value 0 or 1
- s receives 1 and t receives 0
- And the objective function counts the number of arcs starting from a 1-vertex and ending in a 0-vertex

Then a minimimization over all the $\{0, 1\}$ assignments would give a cut of minimum size (justify !). We claim that it is actually possible to make such a representation via the following LP:

$$
\begin{aligned}
z^* = \min & \sum_{a \in A} w_a y_a \\
& \text{under} \\
y_u - y_v - y_a \leq 0 & \text{ for every } a = (u, v) \\
y_s &= 1 \\
y_t &= 0 \\
y_u, y_{uv} &\in \{0, 1\}
\end{aligned}
\tag{2}
$$

We claim that:

- It is the dual of the Max-Flow problem.
- The constraint matrix is TU.

And then the Ford-Fulkerson theorem follows !