Chapter 11

Fractional Relaxation

Lots of combinatorial problems are very close to linear programmes. Indeed they can be formulated as a linear programme with the extra requirement that some of the variable be integers (and not real). Such programmes are called *integer linear programmes*.

For example, consider MAXIMUM MATCHING, which is the problem of finding the largest matching in a graph. Recall that a *matching* in a graph is a set of pairwise non-adjacent edges. The size of a largest matching in a graph G is denoted by $\mu(G)$. With any matching M of a graph G, we may associate its $(0,1)$-valued indicator vector $\mathbf{x}$ (that is $x_e = 1$ if $e \in M$ and $x_e = 0$ otherwise). Since no matching has more than one edge incident with any vertex, every such vector $\mathbf{x}$ satisfies the constraint $\sum_{e \ni v} x_e \leq 1$, for all $v \in V$. Thus an instance of MAXIMUM MATCHING may be formulated as the following integer linear programme.

$$\begin{array}{l} \text{Maximize } \sum_{e \in E} x_e \text{ subject to :} \\ \\ \begin{array}{ll} \sum_{e \ni v} x_e \leq 1 & \text{for all } v \in V \\ x_e \in \mathbb{N} & \text{for all } e \in E \end{array} \end{array} \tag{11.1}$$

For every edge e, the constraint in any of its endvertices implies that $x_e \leq 1$. So the condition $x_e \in \mathbb{N}$ is equivalent to the condition $x_e \in \{0,1\}$.

As a second example, consider VERTEX COVER, the problem of finding a minimum vertex cover in a graph. Recall that a *vertex cover* in a graph $G = (V,E)$ is a set of vertices $S \subset V$ such that every edge has at least one endvertex in S. A vertex cover of minimum cardinality is said to be *minimum*. The cardinality of a minimum vertex cover is denoted by $\nu(G)$. Letting $\mathbf{y}$ be the indicator vector of a vertex cover (i.e. $y_v = 1$ if it is in the vertex cover and $y_v = 0$ otherwise), an instance of VERTEX COVER has following Integer Linear Programming formulation.

$$\begin{array}{l} \text{Minimize } \sum_{v \in V} y_v \text{ subject to :} \\ \\ \begin{array}{ll} y_u + y_v \geq 1 & \text{for all } uv \in E \\ y_v \in \mathbb{N} & \text{for all } v \in V \end{array} \end{array} \tag{11.2}$$

Again the condition $\mathbf{y}$ integral is equivalent to the condition $\mathbf{y}$ $(0,1)$-valued. Indeed if $y_v > 1$ for some v, then setting y_v to 1, we obtain a solution with smaller weight, so an optimal solution is

necessarily $(0,1)$-valued.

VERTEX COVER is known to be NP- hard (see [11]) and, like many NP- hard problems (see e.g. Exercise 11.1 and 11.2), it may be formulated as integer linear programme. Therefore, solving integer linear programmes is NP- hard in general. However, writing a problem as an integer linear programme is often very useful. First, one can always relax the integrality conditions to obtain a linear programme, called *fractional relaxation* of problem.

For instance, the fractional relaxation of (11.1) is the following linear programme.

$$\begin{array}{l} \text{Maximize } \sum_{e\in E} x_e \text{ subject to :} \\ \\ \begin{array}{rl} \sum_{e\ni v} x_e \leq 1 & \text{for all } v \in V \\ x_e \geq 0 & \text{for all } e \in E \end{array} \end{array} \tag{11.3}$$

Since a feasible solution of an integer linear programme is always a feasible solution of its fractional relaxation, the optimal solution of this linear programme is at least as good as that of the integer linear programme. That is, in a maximization problem, the relaxed programme has a value greater than or equal to that of the original programme, while in a minimization problem the relaxed programme has a value smaller than or equal to that of the original programme. For instance, a feasible solution to (11.3), is called a *fractional matching*, and the maximum value of a fractional matching of G is called the *fractional matching number* and is denoted $\mu_f(G)$. Then $\mu_f(G) \geq \mu(G)$. For many graphs G, we have $\mu_f(G) > \mu(G)$. For example, for any odd cycle C_{2k+1}. Setting $x_e = 1/2$ for all edge e, we obtain that $\mu_f(G) \geq (2k+1)/2$ (in fact, $\mu_f(G) = (2k+1)/2$, see Exercise 11.3) while $\mu(G) = k$.

The problem FRACTIONAL MATCHING, which consists in finding a fractional maximal of value $\mu_f(G)$ can be formulated as the linear programme (fractional-matching) which has a polynomial number of constraints. Therefore, it can be solved in polynomial time.

Similarly, the fractional relaxation of (11.2) is

$$\begin{array}{l} \text{Minimize } \sum_{v\in V} y_v \text{ subject to :} \\ \\ \begin{array}{rl} y_u + y_v \geq 1 & \text{for all } uv \in E \\ y_v \geq 0 & \text{for all } v \in V \end{array} \end{array} \tag{11.4}$$

A feasible solution to (11.4), is called a *fractional vertex cover*, and the minimum value of a fractional cover of G is called the *fractional vertex cover number* and is denoted $\nu_f(G)$. Then $\nu_f(G) \leq \nu(G)$. For many graphs G, we have $\nu_f(G) < \nu(G)$. For example, for any odd cycle C_{2k+1}. Setting $y_v = 1/2$ for all vertex v, we obtain that $\nu_f(G) \leq (2k+1)/2$ while $\nu(G) = k+1$. The problem FRACTIONAL VERTEX COVER, which consists in finding a fractional vertex cover of value $\nu_f(G)$ can be solved in polynomial time, because of its linear programming formulation (fractional-cover).

Observe that the two linear programmes (11.3) and (11.4) are dual to each other. Thus by the Duality Theorem, $\mu_f(G) = \nu_f(G)$, and so

$$\mu(G) \leq \mu_f(G) = \nu_f(G) \leq \nu(G).$$

Hence, fractional relaxation and the Duality Theorem prove the (simple) fact that in a graph the maximum size of a matching is smaller than the minimum size of a vertex cover. More generally, it may be used to prove some relation between two parameters which may seem unrelated at first glance, and may lead to nice and elegant proofs. See Subsection 11.1.3.

But fractional relaxation is also very useful in an algorithmic prospect. Indeed sometimes the fractional problem has the same optimal value as the original one, and so solving the former via linear programming yields a polynomial-time algorithm to solve the later. This is the case for MAXIMUM MATCHING in bipartite graphs.

Proposition 11.1. *If G is bipartite, then (11.3) has an integral optimal solution, which is thus an optimal solution to (11.1). So $\mu(G) = \mu_f(G)$.*

Proof. Let $\mathbf{x}$ be an optimal solution to (11.3). Then $\mu_f(G) = \sum_{e \in E} x_e$.

If $\mathbf{x}$ is integral, then we are done. If not, we describe a procedure that yields another optimal solution with strictly more integer coordinates than $\mathbf{x}$. We then reach an integral optimal solution by finitely many repetitions of this procedure.

Let H be the subgraph of G induced by the set of edges $\{e \mid x_e \notin \{0,1\}\}$.

Suppose first that H contains a cycle $C = (v_1, v \ldots, v_k, v_1)$. Since G and so H is bipartite, C must be even.

Let $\varepsilon = \min_{e \in E(C)} \min\{x_e, 1 - x_e\}$. Define $\mathbf{x}'$ and $\mathbf{x}''$ as follows:

$$x'_e = \begin{cases} x_e - \varepsilon, & \text{if } e = v_i v_{i+1},\ 1 \leq i \leq k-1 \text{ and } i \text{ is odd}, \\ x_e + \varepsilon, & \text{if } e = v_i v_{i+1},\ 1 \leq i \leq k \text{ and } i \text{ is even}, \\ x_e, & \text{if } e \notin E(C). \end{cases}$$

and

$$x''_e = \begin{cases} x_e + \varepsilon, & \text{if } e = v_i v_{i+1},\ 1 \leq i \leq k-1 \text{ and } i \text{ is odd}, \\ x_e - \varepsilon, & \text{if } e = v_i v_{i+1},\ 1 \leq i \leq k-1 \text{ and } i \text{ is even}, \\ x_e, & \text{if } e \notin E(Q). \end{cases}$$

where the indices must be considered modulo k. These are two admissible solutions to (11.3). Moreover,

$$\sum_{e \in E} x_e = \frac{1}{2} \left(\sum_{e \in E} x'_e + \sum_{e \in E} x''_e \right).$$

Thus $\mathbf{x}'$ and $\mathbf{x}''$ are also optimal solutions and, by the choice of ε, one of these two solutions has more integer coordinates than $\mathbf{x}$.

Hence, we may assume that H has no cycle. Consider a longest path $P = (v_1, v_2 \ldots, v_k)$ in H. Observe if e is an edge e incident to v_1 (resp. v_k) and different from $v_1 v_2$, (resp. $v_{k-1} v_k$, then $x_e = 0$, for otherwise H would contain either a cycle or a longer path.

Defining $\varepsilon = \min_{e \in E(P)} \min\{x_e, 1 - x_e\}$ and $\mathbf{x}'$ and $\mathbf{x}''$ similarly as above, we obtain two admisible solutions to (11.3). Observe that if P is odd, then the value of $\mathbf{x}''_e$ is greater than the one of $\mathbf{x}$, which contradicts the optimality of $\mathbf{x}$. If P is even, both $\mathbf{x}'$ and $\mathbf{x}''$ are also optimal solutions and, by the choice of ε, one of these two solutions has more integer coordinates than $\mathbf{x}$. □

Remark 11.2. Observe that the proof of Proposition 11.1 easily translates into a polynomial-time algorithm for transforming an optimal solution to (11.3) into an integral optimal solution to it which is necessarily an optimal solution to (11.1). Hence, one can solve MAX MATCHING in bipartite graphs in polynomial time by computing an optimal solution to (11.3) and then modifying it into an optimal solution to (11.1).

One can show the following analogue for VERTEX COVER in bipartite graphs. (Exercise 11.4).

Proposition 11.3. *If G is bipartite, then (11.4) has an integral optimal solution, which is thus an optimal solution to (11.2). So $\nu(G) = \nu_f(G)$.*

Propositions 11.1 and 11.3, together with the Duality Theorem, now imply the following fundamental min–max theorem, due independently to König [20] and Egerváry [9].

Theorem 11.4 (KÖNIG–EGERVÁRY THEOREM)**.**
In any bipartite graph G, the number of edges in a maximum matching is equal to the number of vertices in a minimum vertex cover. In other words, $\mu(G) = \nu(G)$.

In fact, Propositions 11.1 and 11.3 are particular cases of a more general paradigm, called *total unimodularity*, which gives a sufficient condition for a linear programme to have an integral solution. This is discussed in Section 11.1. There are other examples for which the optimal values of the problem and its relaxation are equal. One of them is given in Subsection 11.2.1. For all these problems, solving the fractional relaxation gives a polynomial-time algorithm to solve the problem.

But very often the optimal value of the problem does not equal the optimal value of its relaxation. Moreover, for many problems like VERTEX COVER, we cannot expect any polynomial-time algorithm (unless $\mathcal{P} = \mathcal{NP}$) because they are $\mathcal{NP}$-hard. However, sometimes the optimal values of the probelm and its fractional relaxation are close to each other. In that case, solving the fractional relaxation gives an approximation of the optimal value of the problem. Moreover, one can very often derive an approximate solution of the problem from an optimal solution of the relaxation. The most common technique rounds fractional solutions to integer solutions. Some examples of deterministic and randomized roundings are given in Sections 11.2 and 11.3, respectively.

Finally, for some problems, the optimal value of the fractional relaxation is a very bad estimate for the integer linear programme. This is in particular the case for graph colouring, as shown in Section 11.4.2.

11.1 Total unimodularity

We have seen that the objective function of a linear programme attains its optimum at one of the extreme points of the associated convex polyhedron. A polyhedron is said to be *integral*, if all its extreme points are integral. If the polyhedron $\mathbf{Ax} \leq \mathbf{b}, \mathbf{x} \geq 0$ is integral, then every integer linear programme associated to it has the same optimal value as its fractional relaxation.

Moreover, since the Simplex Algorithm goes from extreme point to extreme point, in such a case it returns an integral solution.

Some integral polyhedra have been characterized by Hoffman and Kruskal [15]. A matrix $\mathbf{A}$ is *totally unimodular* if the determinant of each of its square submatrices is equal to 0, $+1$, or -1. Using Cramér's rule, it is easy linear algebra to show the following.

Theorem 11.5 (Hoffman and Kruskal [15])**.** *The polyhedron defined by* $\mathbf{Ax} \leq \mathbf{b}, \mathbf{x} \geq 0$ *is integral for every integral vector* $\mathbf{b}$ *if and only if* $\mathbf{A}$ *is a totally unimodular matrix.*

In particular, if $\mathbf{A}$ *is totally unimodular and* $\mathbf{b}$ *is an integral vector, then the linear programme*

$$\textit{Maximize} \quad \mathbf{c^T x} \quad \textit{subject to :}$$

$$\begin{array}{rcl} \mathbf{Ax} & \leq & \mathbf{b} \\ \mathbf{x} & \geq & 0 \end{array}$$

has an integral optimal solution (if it has one).

Remark 11.6. This characterization requires $\mathbf{b}$ to vary. For a given vector $\mathbf{b}$ it may be true that $\{\mathbf{x} \mid \mathbf{Ax} \leq \mathbf{b}, \mathbf{x} \geq 0\}$ is an integral polyhedron even if $\mathbf{A}$ is not totally unimodular.

Many totally unimodular matrices have been known for a long time.

Theorem 11.7 (Poincaré [24])**.** *Let* $\mathbf{A}$ *be a* $(0,-1,+1)$*-valued matrix, where each column has at most one* $+1$ *and at most one* -1*. Then* $\mathbf{A}$ *is totally unimodular.*

Proof. Let $\mathbf{B}$ be a k by k submatrix of $\mathbf{A}$. If $k = 1$, then $\det(\mathbf{B})$ is either 0, $+1$ or -1. So we may suppose that $k \geq 2$ and proceed by induction on k. If $\mathbf{B}$ has a column having at most one non-zero, then expanding the determinant along this column, we have that $\det(\mathbf{B})$ is equal to 0, $+1$, or -1, by our induction hypothesis. On the other hand, if every column of $\mathbf{B}$ has both a $+1$ and a -1, then the sum of the rows of $\mathbf{B}$ is 0 and so $\det(\mathbf{B}) = 0$. □

Seymour [25] gave a characterization of totally unimodular matrices, from which a polynomial-time algorithm to recognize such matrices follows.

In the following two subsections, we show how celebrated min-max theorems follows from total unimodularity. We then give another application of total unimodularity.

11.1.1 Matchings and vertex covers in bipartite graphs

We now show how Propositions 11.1 and 11.3 follows total unimodularity.

Let $G = (V, E)$ be a graph. The *incidence matrix* $\mathbf{A}$ of G is the matrix whose rows are indexed by V and the columns by E, and whose entries are defined by

$$a_{v,e} = \begin{cases} 1 & \text{if } v \text{ is incident to } e \\ 0 & \text{otherwise} \end{cases}$$

Hence, the matricial form of (11.3) and (11.4) are, respectively,

$$\text{Maximize } \mathbf{1}^T\mathbf{x} \text{ subject to } \mathbf{Ax} \leq \mathbf{1} \text{ and } \mathbf{x} \geq \mathbf{0}. \tag{11.5}$$

and

$$\text{Minimize } \mathbf{1}^T\mathbf{y} \text{ subject to } \mathbf{A}^T\mathbf{y} \geq \mathbf{1} \text{ and } \mathbf{y} \geq \mathbf{0}. \tag{11.6}$$

where $\mathbf{A}$ is the incidence matrix of G.

Proposition 11.8. *The incidence matrix of a bipartite graph is totally unimodular*

Proof. Left as Exercise 11.5. □

Proposition11.8 and Theorem 11.5 imply that (11.5) and (11.6) have integral optimal solutions. This is Propositions 11.1 and 11.3.

11.1.2 Flows via linear programming

The Maximum Flow Problem (7.3) can be formulated as a linear programme. Indeed writing f_{uv} instead of $f(u,v)$ and c_{uv} instead of $c(u,v)$, the problem becomes

$$\begin{array}{lll} \text{Maximize} & \displaystyle\sum_{v\in N^+(s)} f_{sv} & \\ \text{Subject to:} & & \\ & f_{uv} \leq c_{uv} & \text{for all } uv \in E \\ & \displaystyle\sum_{u\in N^-(v)} f_{uv} - \sum_{w\in N^+(v)} f_{vw} = 0 & \text{for all } v \in V\setminus\{s,t\} \\ & f_{uv} \geq 0 & \text{for all } uv \in E \end{array}$$

Let $D = (V,E)$ be a digraph. The *incidence matrix* $\mathbf{A}$ of G is the matrix whose rows are indexed by V and the columns by E, and whose entries are defined by

$$a_{v,e} = \begin{cases} +1 & \text{if } v \text{ is the head of } e \\ -1 & \text{if } v \text{ is the tail of } e \\ 0 & \text{otherwise} \end{cases}$$

Let $\mathbf{A}'$ be the matrix obtained from $\mathbf{A}$ by removing the rows corresponding to s and t, $\mathbf{I}$ be the $|E|$ by $|E|$ identity matrix, $\mathbf{d}$ be the indicator vector of set of arcs leaving s, and $\mathbf{c} = (c_e)_{e\in E}$. Then a possible matricial formulation of the Maximum Flow Problem is

$$\text{Maximize } \mathbf{d}^T\mathbf{f} \text{ subject to } \mathbf{If} \leq \mathbf{c} \text{ and } \mathbf{A}'\mathbf{f} = \mathbf{0} \text{ and } \mathbf{f} \geq \mathbf{0}. \tag{11.7}$$

Similarly to Theorem 11.7, one can show that the matrix $\mathbf{M} := \begin{bmatrix} \mathbf{I} \\ \mathbf{A'} \end{bmatrix}$ is unimodular (Exercise 11.6). Thus Theorem 11.5 implies that, if all capacities are integers, then the maximum value of a flow is an integer and that there exists an integral maximum flow. Moreover, as already obeserved, the Simplex Algorithm returns an integral flow. Such a flow is also returned by Ford–Fulkerson Algorithm (See Remark 7.12).

The total unimodularity also implies Ford–Fulkerson Theorem (7.7). *The maximum value of an (s,t)-flow equals the minimum capacity of an (s,t)-cut.*.

Alternative proof of Theorem 7.7. By Exercise 9.15, the dual problem of (11.7) is

$$\text{Minimize } \mathbf{c}^T\mathbf{y} \text{ subject to } \mathbf{y}+\mathbf{A'}^T\mathbf{z} \geq \mathbf{d} \text{ and } \mathbf{y} \geq \mathbf{0}. \tag{11.8}$$

Because the matrix $[\mathbf{I}, \mathbf{A'}^T]$ is totally unimodular, by Theorem 11.5, the minimum of (11.8) is attained by integral vectors $\tilde{\mathbf{y}}$ and $\tilde{\mathbf{z}}$.

Now define $V_s = \{v \in V \setminus \{s,t\} \mid \tilde{z}_v < 0\} \cup \{s\}$ and $V_t = V \setminus V_s$. Then $C = (V_s, V_t)$ is an (s,t)-cut. We shall prove that C is a cut with capacity less or equal to the maximum flow value $\nu_{\max}$. By the Duality Theorem (9.10), $\nu_{\max} = \mathbf{c}^T\tilde{\mathbf{y}}$, and trivially $\delta(C) \geq \nu_{\max}$, we need to prove that $\delta(C) \leq \mathbf{c}^T\tilde{\mathbf{y}}$. But $\delta(C) = \sum_{e\in C} c(e)$ and $\mathbf{c}^T\tilde{\mathbf{y}} = \sum_{e\in E} c(e)\tilde{y}_e$. Since the $\tilde{y}_e$ are non-negative, it is sufficient to prove that $\tilde{y}(e) \geq 1$ for all $e \in C$.

Let uv be an arc of C. Recall that $\tilde{\mathbf{y}} + \mathbf{A'}^T\tilde{\mathbf{z}} \geq \mathbf{d}$. If $u \neq s$, then $\tilde{y}_e + \tilde{z}_u - \tilde{z}_v \geq 0$,with $\tilde{z}_t = 0$. But by definition of V_s, $\tilde{z}_v \geq 0$ and $\tilde{z}_u < 0$, and so $\tilde{z}_u \leq -1$, since $\tilde{\mathbf{z}}$ is integral. Hence, $\tilde{y}_e \geq 1$. If $u = s$, then $\tilde{y}_e - \tilde{z}_v \geq 1$, which again implies $\tilde{y}_e \geq 1$. □

11.1.3 Covering a strong digraph with directed cycles

Gallai–Milgram Theorem (6.14) states that every digraph D can be covered by at most $\alpha(D)$ directed paths. A natural question is to ask if a digraph could be covered by few directed cycles. In general, the answer is no as there are digraphs in which some vertices are in no cycles. But in every strong digraph, any vertex is contained in a directed cycle. In 1964, Gallai [10] conjectured an analogue of Gallai–Milgram Theorem for covering strong digraphs with directed cycles. This was proved in 2004 by Bessy and Thomassé [4].

Theorem 11.9 (Bessy and Thomassé [4])**.** *The vertex set of any non-trivial strong digraph D can be covered by $\alpha(D)$ directed cycles.*

They established it by proving a stronger result, namely a min–max theorem relating a cyclic analogue of the stability number to the minimum index of a cycle covering. Here, we present a closely related min–max theorem established by Bondy, Charbit and Sebö.

Let $D = (V,E)$ be a digraph. By a *cyclic order* of D we mean a cyclic order $O = (v_1, v_2, \ldots, v_n, v_1)$ of its vertex set V. Given such an order O, each directed cycle of D can be thought of as winding around O a certain number of times. In order to make this notion precise, we define the *length* of an arc (v_i, v_j) of D (with respect to O) to be $j-i$ if $i < j$ and $n+j-i$ if $i > j$. Informally, the length of an arc is just the length of the segment of O 'jumped' by the arc. If C is a directed cycle of D, the sum of the lengths of its arcs is a certain multiple of n. This multiple is called the *index* of C (with respect to O), and denoted $i(C)$. By extension, the *index* of a family $\mathcal{C}$ of directed cycles, denoted $i(\mathcal{C})$, is the sum of the indices of its constituent cycles.

A *weighting* of the vertices of a digraph D is a function $w : V \to \mathbb{N}$. We refer to $w(v)$ as the *weight* of vertex v. By extension, the *weight* $w(H)$ of a subgraph H of D is the sum of the weights of its vertices. If D is equipped with a cyclic order O, and if $w(C) \leq i(C)$ for every directed cycle C of D, we say that the weighting w is *index-bounded* (with respect to O). Observe that for any cycle covering $\mathcal{C}$ of D and any index-bounded weighting w,

$$i(\mathcal{C}) \geq \sum_{C\in\mathcal{C}} w(C) \geq w(D). \tag{11.9}$$

Theorem 11.10. *Let D be a digraph each of whose vertices lies in a directed cycle, and let O be a cyclic order of D. Then:*

$$\min i(\mathcal{C}) = \max w(D) \tag{11.10}$$

where the minimum is taken over all cycle coverings $\mathcal{C}$ of D and the maximum over all index-bounded weightings w of D.

In order to deduce Theorem 11.9 from Theorem 11.10, it suffices to apply it to a coherent cyclic order O of D. A cyclic order is *coherent* if every arc lies in a directed cycle of index one. Bessy and Thomassé [4] showed that every strong digraph admits a coherent cyclic order. A fast algorithm for finding coherent cyclic orders can be found in [16].

We then observe that:

- for every family $\mathcal{C}$ of directed cycles of D, we have $|\mathcal{C}| \leq i(\mathcal{C})$,
- because each vertex lies in a directed cycle and O is coherent, each vertex lies in a simple cycle, so an index-bounded weighting of D is necessarily $(0,1)$-valued,
- because each arc lies in a simple cycle, in an index-bounded weighting w no arc can join two vertices of weight one, so the support of w is a stable set, and $w(D) \leq \alpha(D)$.

Proof of Theorem 11.10. Let D be a digraph with a cyclic order $(v_1, \ldots, v_n, v_1)$.

An arc (v_i, v_j) is called a *forward arc* of D if $i < j$, and a *reverse arc* if $j < i$. Observe that for every cycle C, $i(C)$ is equals the number of reverse arcs in C.

Consider a collection of cycles $\mathcal{C}$ and for every arc e let x_e be the number of cycles of $\mathcal{C}$ containing e. Then $\mathbf{x}$ is a *circulation* that is $\sum_e \text{entering } {}_v x_e = \sum_e \text{leaving } {}_e x_e$ for each vertex v. And reciprocally every integral circulation corresponds to a collection of cycles. (See Exercise 11.8).

Now a circulation $\mathbf{x}$ corresponds to a covering if for every vertex v there is an arc e leaving v for which $x_e \geq 1$, which is equivalent to $\sum_e \text{leaving } {}_e x_e \geq 1$.

Denoting by $\mathbf{M}$ be incidence matrix of D and $\mathbf{N}$ is the $n \times m$ matrix defined by:

$$n_{ve} = \begin{cases} 1 & \text{if } v \text{ is the tail of } e \\ 0 & \text{otherwise} \end{cases}$$

finding the minimum index of a cycle covering of D can be written as the linear programme :

$$\text{Minimize } \mathbf{c}^T\mathbf{x} \text{ subject to } \mathbf{Mx} = \mathbf{0} \text{ and } \mathbf{Nx} \geq \mathbf{1} \text{ and } \mathbf{x} \in \mathbb{N}. \tag{11.11}$$

where $\mathbf{c}$ indicates if e is reverse, that is $c_e = 1$ if e is reverse and $c_e = 0$ otherwise.

Let us show that $\mathbf{Q} := \begin{bmatrix} \mathbf{M} \\ \mathbf{N} \end{bmatrix}$ is totally unimodular. Consider the matrix $\widetilde{\mathbf{Q}}$ obtained from $\mathbf{Q}$ by subtracting each row of $\mathbf{N}$ from the corresponding row of $\mathbf{M}$. Each column of $\widetilde{\mathbf{Q}}$ contains one 1 and one -1, the remaining entries being 0. Thus, by Theorem 11.7, $\widetilde{\mathbf{Q}}$ is totally unimodular. Because $\widetilde{\mathbf{Q}}$ was derived from $\mathbf{Q}$ by elementary row operations, the matrix $\mathbf{Q}$ is totally unimodular too.

The linear programme (11.11) is feasible because, by assumption, D has at least one cycle covering, and it is bounded because $\mathbf{c}$ is non-negative. Thus (11.11) has an optimal solution.

By Theorem 11.5, the optimal value of (11.11) is equal to the optimal value of its fractional relaxation, which in turn by the Duality Theorem is equal to the optimal value of its dual

$$\text{Maximize } \mathbf{1}^T\mathbf{z} \text{ subject to } \mathbf{M}^T\mathbf{y}+\mathbf{N}^T\mathbf{z} \leq \mathbf{d} \text{ and } \mathbf{z} \geq \mathbf{0}. \tag{11.12}$$

Then (11.12) is the problem of maximizing $\sum_{v\in V} z_v$ subject to the constraints:

$$y_u - y_v + z_u \leq \begin{cases} 1 & \text{if } e=(u,v) \text{ is a reverse arc} \\ 0 & \text{if } e=(u,v) \text{ is a forward arc} \end{cases}$$

By Theorem 11.5, this problem has an integral optimal solution to (11.12). Consider an integal feasible solution. If we sum the above constraints over the arc set of a directed cycle C of D, we obtain the inequality

$$\sum_{v\in V(C)} z_v \leq i(C)$$

In other words, the function w defined by $w(v) := z_v$, for all $v \in V$, is an index-bounded weighting, and the objective function is the weight $w(D)$ of D. Hence the optimal value of (11.12) is at most the maximum of $w(D)$ over all index-bounded weightings w of D.

Hence min $i(\mathcal{C}) \leq$ max $w(D)$. But by (11.9), min $i(\mathcal{C}) \geq$ max $w(D)$, so min $i(\mathcal{C}) =$ max $w(D)$. □

11.2 Deterministic rounding

Unfortunately, most of linear programmes do not have a totally unimodular constraint matrix. However, the fractional relaxation may still be of interest. It sometimes leads to polynomial-time algorithms to solve the problem either optimally or approximately. The general idea is to first compute a solution of the fractional relaxation and then derive from this solution a solution to the original problem which is either optimal or near optimal.

11.2.1 Minimum cut

Recall that an (s,t)-cut in a graph is a bipartition (V_s,V_t) with $s \in V_s$ and $t \in V_t$. It is completely determined by the indicator vector $\mathbf{p}$ of V_t. Hence a formulation of the minimum (s,t)-cut in a graph $G=(V,E)$ is the following.

$$\text{Minimize} \quad \sum_{uv\in E} c_{uv}|p_u - p_v| \quad \text{subject to :}$$

$$\begin{array}{rcll} p_s & = & 0 & \\ p_t & = & 1 & \\ p_v & \in & \{0,1\} & \forall v \in V(G) \end{array} \tag{11.13}$$

It does not immediately appear to be an integer linear programme, but it indeed is for it is equivalent to the following.

$$\begin{array}{lll} \text{Minimize} \quad \sum_{uv\in E} c_{uv}q_{uv} & \text{subject to :} & \\ \\ q_{uv} - p_u + p_v \;\geq 0 & \forall uv \in E & \\ q_{uv} - p_v + p_u \;\geq 0 & \forall uv \in E & \\ p_s \;=\; 0 & & \\ p_t \;=\; 1 & & \\ p_v \;\in\; \{0,1\} & & \forall v \in V \\ q_{uv} \;\in\; \{0,1\} & \forall uv \in E & \end{array} \tag{11.14}$$

One can check that the matrix corresponding to (11.14) is not totally unimodular (Exercise 11.10). However, solving the fractional relaxation of this problem, (where the constraint $p_v \in \{0,1\}$ is replaced by $0 \leq p_v \leq 1$), gives us the value of a minimum cut and, almost immediately, an optimal solution.

Proposition 11.11. *Let* $\mathbf{p}$ *be an optimal solution of the fractional relaxation of (11.13). Then for all* $0 \leq \gamma < 1$*, the* $(0,1)$*-valued vector* $\mathbf{p}^\gamma$ *defined by* $p_u^\gamma = 0$*, if* $p_u < \gamma$ *and* $p_u^\gamma = 1$ *otherwise, is an optimal an optimal solution of (11.13) and its fractional relaxation.*

Proof. We prove it by induction on the number k of values other than 0 and 1 taken the p_v's, the result holding trivially if $k = 0$. Suppose now that $k \geq 1$. Let $0 = p_0 < p_1 < p_2 < \cdots < p_k < p_{k+1} = 1$ be the values taken by the p_v's.

Assume that $p_1 < \gamma$. Let P (resp. L, R) be the set of vertices v such that $p_v = p_1$ (resp. $p_v = 0$, $p_v > p_1$).

$$\sum\{c_{vu} \mid u \in P, v \in L, uv \in E\} = \sum\{c_{vu} \mid u \in P, v \in R, uv \in E\} \tag{11.15}$$

for otherwise one of $\mathbf{p}'$ and $\mathbf{p}''$ defined by

$$p'_v = \begin{cases} p_v & \text{if } v \in V \setminus P \\ 0 & \text{if } v \in P \end{cases} \quad \text{and} \quad p''_v = \begin{cases} p_v & \text{if } v \in V \setminus P \\ p_2 & \text{if } v \in P \end{cases}$$

would contradict that $\mathbf{p}$ is an optimal solution. Furthermore, because of (11.15), $\mathbf{p}'$ (and also $\mathbf{p}''$) is an optimal solution which takes one value less than $\mathbf{p}$. Hence by induction, $\mathbf{p}^\gamma$ is an optimal solution of (11.13) and its fractional relaxation.

A symmetrical argument yields the result if $p_k > \gamma$. □

11.2.2 Vertex cover: 2-approximation and kernel.

If a graph is non- bipartite, then its incidence matrix is not totally unimodular (Exercise 11.5). We have also seen that for the odd cycles the fractional vertex cover number is strictly less than the vertex cover number. Moreover finding an optimal vertex cover is NP-hard, so we cannot hope to have a polynomial-time algorithm to find one. However we shall now see that there is a 2-approximate algorithm by solving FRACTIONAL VERTEX COVER and then deriving a solution to VERTEX COVER.

Theorem 11.12. FRACTIONAL VERTEX COVER *has an optimal solution which is half-integral, that is* $(0,1/2,1)$*-valued.*

Proof. Let $\mathbf{y}$ be an optimal solution of FRACTIONAL VERTEX COVER with the largest number of coordinates in $\{0,1/2,1\}$.

Suppose that $\mathbf{y}$ does not have all its coordinates in $\{0,1/2,1\}$. Set $\varepsilon = \min\{y_v, |y_v - \frac{1}{2}|, 1 - y_v \mid v \in V \text{ and } y_v \notin \{0,1/2,1\}\}$.

Consider $\mathbf{y}'$ and $\mathbf{y}''$ defined as follows:

$$y'_v = \begin{cases} y_v - \varepsilon, & \text{if } 0 < y_v < \frac{1}{2}, \\ y_v + \varepsilon, & \text{if } \frac{1}{2} < y_v < 1, \\ y_v, & \text{otherwise.} \end{cases} \quad \text{and} \quad y''_v = \begin{cases} y_v + \varepsilon, & \text{if } 0 < y_v < \frac{1}{2}, \\ y_v - \varepsilon, & \text{if } \frac{1}{2} < y_v < 1, \\ y_v, & \text{otherwise.} \end{cases}$$

These are two admissible solutions to FRACTIONAL VERTEX COVER. Moreover, $\sum_{v\in V} y_v = \frac{1}{2}\left(\sum_{v\in V} y'_v + \sum_{v\in V} y''_v\right)$. Thus $\mathbf{y}'$ and $\mathbf{y}''$ are also optimal solutions and, by the choice of ε, one of these two solutions has more coordinates in $\{0,1/2,1\}$ than $\mathbf{y}$, a contradiction. □

Remark 11.13. Observe that the proof of Theorem 11.12 easily translates into a polynomial-time algorithm for transforming an optimal solution to FRACTIONAL VERTEX COVER into an half-integral optimal solution to it.

Corollary 11.14. *There is an algorithm which returns a* 2*-approximate solution to* VERTEX COVER *in polynomial time.*

Proof. The algorithm is very simple. We first solve FRACTIONAL VERTEX COVER and derive an half-integral optimal solution $\mathbf{y}^f$ to it. Define $\mathbf{y}$ by $y_v = 1$ if and only if $y^f_v \in \{1/2;1\}$. Clearly, $\mathbf{y}$ is an admissible solution of VERTEX COVER. Moreover, by definition

$$\sum_{v\in V} y_v \leq 2 \sum_{v\in V} y^f_v = 2 \cdot \nu^f(G) \leq 2 \cdot \nu(G).$$

□

No better constant-factor approximation algorithm than the above one is known. The minimum vertex cover problem is APX-complete, that is, it cannot be approximated arbitrarily well unless $\mathcal{P} = \mathcal{NP}$. Using techniques from the PCP theorem, Dinur and Safra [7] proved that VERTEX COVER cannot be approximated within a factor of 1.3606 for any sufficiently large vertex degree unless $\mathcal{P} = \mathcal{NP}$. Moreover, if the unique games conjecture is true, then VERTEX COVER cannot be approximated within any constant factor better than 2 as shown by Khot and Regev [18].

Analysing in more details relations between the solutions of FRACTIONAL VERTEX COVER and those of VERTEX COVER, we can find a kernelization of the following parameterized version of the vertex cover problem.

Problem 11.15 (PARAMETERIZED VERTEX COVER)**.**
Instance: a graph G and an integer k.
Parameter: k.
Question: does G have a vertex cover of cardinality at most k ? In other words, $\nu(G) \leq k$?

In the area of parameterized algorithms, one of the major issue is to determine if a problem is FPT (for Fixed-Parameter Tractable), that is if it can be solved by an algorithm in time $f(k)P(n)$ with f an arbitrary function in k and P a polynomial in n, the number of vertices of G. One of the possible ways to do so, is to prove that there exists a $g(k)$-*kernelization* for some function $g(k)$, that is a polynomial-time algorithm that transforms any instance (G,k) of the problem into an equivalent instance (G',k') such that $|G'| \leq g(k)$ and $k' \leq k$. Once we know that a problem admits a kernelization, a second issue is to find the minimum function g for which it has a $g(k)$-kernelization.

In the remaining of this subsection, we shall prove that PARAMETERIZED VERTEX COVER has a $2k$-kernelization.

Let $\mathbf{y}^f$ be a half-integral optimal solution to FRACTIONAL VERTEX COVER. For $t \in \{0,1/2,1\}$, set $V_t = \{v \in V \mid y_v^f = t\}$, $G_t = G\langle V_t \rangle$.

Theorem 11.16 (Nemhauser and Trotter [23])**.** *Let $\mathbf{y}^f$ be a half-integral optimal solution to* FRACTIONAL VERTEX COVER. *Then there exists a minimum cover S of G such that:*

(a) $V_0 \cap S = \emptyset$*;*

(b) $V_1 \subseteq S$.

Proof. We present here a proof due to Khuller [19]. Let us first show that (a) implies(b). Suppose that there exists $v \in V_1 \setminus S$.

- If v has no neighbour in V_0, then one can decrease the weight y_v^f at v to obtain a fractional vertex cover of G with smaller weight, a contradiction to the minimality of $\mathbf{y}^f$.
- If v has a neighbour w in V_0, then by (a) w is not in S so vw is not covered by S, a contradiction.

Let us now prove (a).
Let S be a minimum vertex cover which has the fewest vertices in V_0. Suppose for a contradiction that $S \cap V_0 \neq \emptyset$. Observe that V_0 is a stable set because $\mathbf{y}^f$ is a fractional vertex cover. Also there is no edges between $V_0 \setminus S$ and $V_1 \setminus S$. Suppose moreover that $|V_0 \cap S| < |V_1 \setminus S|$. Then we can increase y_v^f of some tiny ε for all $v \in V_0 \cap S$ and decrease y_v of ε for all $v \in V_1 \setminus S$. This results in a fractional vertex cover of G with weight less than the one of $\mathbf{y}^f$, a contradiction.

Hence we have $|V_0 \cap S| \geq |V_1 \setminus S|$. Thus $(S \setminus V_0) \cup V_1$ is a vertex cover of G with at most as many vertices as S and no vertex in V_0. □

Corollary 11.17. PARAMETERIZED VERTEX COVER *has a $2k$-kernelization.*

Proof. The kernelization algorithm is the following.

Algorithm 11.1.

1. Find an optimal solution $\mathbf{y}^f$ to FRACTIONAL VERTEX COVER.
2. If the weight of $\mathbf{y}^f$ is greater than k, then return a 'No'-instance.
3. If $|V_1| = k$ and $V_{1/2} = \emptyset$, return a 'Yes'-instance.
4. Otherwise return $(G_{1/2}, k - |V_1|)$.

This algorithm clearly runs in polynomial time. In addition, by Theorem 11.16, G has a vertex cover of cardinality at most k if and only if $G_{1/2}$ has a vertex cover of cardinality at most $k - |V_1|$.

The trivial instances returned at Steps 2 or 3 being trivially of size at most $2k$, it remains to prove that the graph $G_{1/2}$ has order at most $2k$. But because of Step 2,

$$k \geq \nu^f(G) \geq \sum_{v \in V} y_v^f = \frac{1}{2}|V_{1/2}| + |V_1|$$

so $|V_{1/2}| \leq 2(k - |V_1|)$. □

11.3 Randomized rounding

11.3.1 Maximum satisfiability

There is a natural optimization version of the Boolean Satisfiability Problem (SAT) (Problem 3.5). Given a Boolean formula F in conjunctive normal form and a non-negative weight w_i associated with each clause C_i, the objective is to find a Boolean assignment to the variables that maximizes the total weight of the satisfied clauses. This problem, called MAX SAT, is clearly NP-hard.

Johnson [17] demonstrated a simple $\frac{1}{2}$-approximation algorithm. It is based on the following simple random assignment: set each variable uniformly at random to *true* or *false*, independently from the other variables. Let $|C_i|$ denote the number of literals in clause C_i. It is easy to check that

$$p_1(C_i) = \mathbf{Pr}(C_i \text{ is satisfied}) = 1 - 2^{-|C_i|}.$$

Hence, the expected weight of clauses satisfied by this random assignment is

$$\mathbf{E}(W) = \sum_{C_i} w_i(1 - 2^{-|C_i|}) \leq \frac{1}{2}\sum_{C_i} w_i.$$

The probabilistic method specifies that there must exist a truth assignment whose weight is at least this expected value. In fact, the method of conditional probabilities (See Chapter 15 of

[1]) can be applied to find such assignment deterministically in polynomial time. In the method of conditional probabilities, the value for thr ith variable is determined in the ith iteration: given the values of $x_1,\ldots,x_{i-1}$ calculate the expected weight of clauses satisfied by the probabilistic assignment, given the current assignment to $x_1,\ldots,x_{i-1}$ and the assignment $x_i = 1$. Then calculate the expected weight given the assignment to $x_1,\ldots,x_{i-1}$ and $x_i = 0$. The variable x_i is assigned the value that maximizes the conditional expectation. Since each conditional expectation can be calculated in polynomial time, the overall algorithm takes polynomial time, and as asserted above, the produced assignment has weight at least $\mathbf{E}(W)$.

Observe that the above algorithm is also a $(1-2^{-k})$-approximation algorithm when each clause contains at least k literals. In particular, if $k \geq 2$, the performance guarantee is at least $\frac{3}{4}$. A general $\frac{3}{4}$-approximation algorithm was proposed by Yannakakis [26]. This algorithm transforms a MAX SAT instance into an equivalent instance (in terms of approximability) which does not contain any clauses with only one literal. In conjunction with Johnson's algorithm, this leads to the performance guarantee of $3/4$. The algorithm uses maximum flow calculation in an elegant way to transform instance in which all clauses have two literals. However, the transformation becomes more complicated when general clauses are introduced. In the next paragraph, we describe a simpler algorithm due to Goemans and Williamson with the same approximation ratio [12]. Asano[3] gave a polynomial-time algorithm with the better approximation ratio 0.77. On the opposite, Håstad [14] proved that, unless $\mathcal{NP} = \mathcal{P}$, the MAX SAT problem cannot be approximated in polynomial time within a ratio greater than $7/8$.

Goemans–Williamson algorithm

The idea is to consider two different randomized procedures for constructing the Boolean assignment, and observe that they have complementary strengths in terms of their approximation guarantees. The first one is Johnson's algorithm and thus works well when each clause has 'many' literals. The second one will be good if each clause has 'few' literals. Thus, we could run both and take the better solution; in particular, we could choose one of the two schemes uniformly at random, and the expectation of the value of the resulting solution will be the arithmetic mean of the expected values of the solutions of the two procedures.

Let us now present and analyze the second procedure. It starts with an integer linear programming formulation. For each clause C_i, let $P(i)$ denote the set of unnegated variables appearing in it, and $N(i)$ be the set of negated variables in it. For each variable j, let $x_j = 1$ if this variable is set to *true*, and $x_j = 0$ if the variable is set to $false$. Letting $z_i \in \{0,1\}$ be the indicator for clause C_i getting satisfied, we obtain the following formulation.

$$\begin{aligned} &\text{Maximize} \quad \textstyle\sum_i w_i z_i \quad \text{subject to :} \\ z_i \quad &\leq \quad \textstyle\sum_{j\in P(i)} x_j + \sum_{j\in N(i)} (1-x_j) \qquad \text{for all variable } i \end{aligned} \tag{11.16}$$

We first solve fractional relaxation of (11.16) obtained by relaxing each x_j and z_i to be a real in $[0,1]$ and consider a optimal solution $(\mathbf{x}^f, \mathbf{z}^f)$ of it. We interpret each x_j^f as a probability. Thus, our randomized rounding process will be, independently for each j, to set $x_j = 1$ (i.e., make variable j *true*) with probability x_j^f and $x_j = 0$ (make variable j $false$) with probability

$1-x_j^f$. One intuitive justication for this is that if x_j^f were 'high', i.e., close to 1, it may be taken as an indication by the linear programme that it is better to set variable j to *true*; similarly for the case where x_j is close to 0.

Let us lower-bound the probability of clause C_i getting satisfied. Without loss of generality, we can assume that all variables appear unnegated in C_i. Thus, we have

$$z_i^f = \min\{s_i^f, 1\} \text{ with } s_i^f = \sum_{j \in P(i)} x_j^f.$$

It is not hard to check that $\mathbf{Pr}(C_i \text{ is satisfied}) = 1 - \prod_{j \in P(i)}(1-x_j^f)$ is minimized when each x_j^f equals $s_i^f/|C_i|$. Thus

$$p_2(C_i) = \mathbf{Pr}(C_i \text{ is satisfied}) \geq 1 - \left(1 - \frac{s_i^f}{|C_i|}\right)^{|C_i|} \geq 1 - \left(1 - \frac{z_i^f}{|C_i|}\right)^{|C_i|}.$$

For a fixed value of z_i^f, the term $1-(1-z_i^f/|C_i|)^{|C_i|}$ decreases monotonically as $|C_i|$ increases. This is the sense in which this scheme is complementary to Johnson's.

So, as mentioned above, suppose we choose one of the two schemes uniformly at random, in order to balance their strengths. Then,

$$\begin{aligned} \mathbf{Pr}(C_i \text{ is satisfied}) &= \frac{1}{2}(p_1(C_i) + p_2(C_i)) \\ &\geq 1 - \frac{1}{2}\left(2^{-|C_i|} + (1 - z_i^f/|C_i|)^{|C_i|}\right) \\ &\geq \tfrac{3}{4} z_i^f \end{aligned}$$

because $z_i^f \in [0,1]$. Indeed for any fixed positive integer k, $f_k(z) = \frac{1}{2}\left(2^{-k} + (1-z/k)^k\right) - \frac{3z}{4}$ has a non-positive derivative for $z \in [0,1]$. Thus, it suffices to show that $f_k(1) \geq 0$ for all positive integer k. We have $f_1(1) = f_2(1) = 0$. For $k \geq 3$, $2^-k \leq 1/8$ and $(1-1/k)^k \leq 1/e$. So $f_k(1) \geq 0$ for $k \geq 3$.

Thus the expected value of of the produced assignment is at least $3/4$ of the optimal value of the fractional relaxation of (11.16), and so at least at least $3/4$ of the optimal value of (11.16) itself.

The method of conditional probabilities may also be used to derandomized the second procedure and thus to get a deterministic $\frac{3}{4}$-approximation algorithm for MAX SAT.

11.3.2 Multiway cut

Let (G,p) be an edge-weigthed graph and let $s_1,\ldots,s_k$ be k vertices called *terminals* $s_1,\ldots,s_k$. An $(s_1,\ldots,s_k)$-*cut* is a partition $\Pi = (V_1,\ldots,V_k)$ of V such that $s_i \in V_i$ for all $1 \leq i \leq k$. This notion generalizes the one of (s,t)-cut defined in Section 7.3 and Subsection 11.2.1.

Let $\Pi = (V_1,\ldots,V_k)$ be a partition of G. An edge e is Π-*transversal* if its endvertices are in different parts. In other words, $e = uv$, $u \in V_i$, $v \in V_j$ and $i \neq j$. The set of transversal edges of

Π is denoted $E(\Pi)$. The *weight* of a cut Π is the sum of the weights of the Π-transversal edges: $w(\Pi) = \sum_{e \in E(\Pi)} w(e)$.

The MULTIWAY CUT problem is the following:

Problem 11.18 (MULTIWAY CUT)**.**
Instance: an edge-weighted graph (G,w) and k vertices $s_1,\ldots,s_k$.
Find: an $(s_1,\ldots,s_k)$-cut of minimum weight.

For $k=2$, it is the problem of finding a minimum-weight (s,t)-cut which can be solved in polynomial time as we saw in Chapter 7 and in Subsection 11.2.1.

For $k \geq 3$, the problem is $\mathcal{NP}$-hard [6]. But, it can be approximated in polynomial time. As shown by Dalhaus et al. [6], one can obtain a $2-\frac{2}{k}$-approximated solution for MULTIWAY CUT by running k times the algorithm for finding a minimum-weight (s,t)-cut. Indeed, for all i, consider the graph G_i obtained from G by identifying the s_j, $j \neq i$ in one vertex t_i and find a minimum-weight (s_i,t_i)-cut C_i in G_i. Then each C_i separates s_i from all other s_j in G. Hence the union of the $k-1$ cuts C_i with smallest weight is an $(s_1,\ldots,s_k)$-cut with weight at most $2-\frac{2}{k}$ times the minimum weight of an $(s_1,\ldots,s_k)$-cut. See Exercise 11.13.

A 3/2-approximation algorithm

Given an $(s_1,\ldots,s_k)$-cut $(V_1,\ldots,V_k)$, for each vertex v, we consider the vector x_v whose ith coordinate is 1 if $v \in V_i$ and 0 otherwise. Clearly, $v \in V_i$ if and only if $x_v = e_i$, where e_i the vector, all coordinates of which are 0 except the i^{th} which is 1. Hence there is a one-to-one correspondence between the vectors $\mathbf{x}$ in $\{e_i \mid 1 \leq i \leq k\}^{|V|}$ such that $x_{s_i} = e_i$ for $1 \leq i \leq k$ and the $(s_1,\ldots,s_k)$-cuts. To each $\mathbf{x}$ corresponds the cut $(V_1,\ldots,V_k)$ defined by $V_i = \{v \mid x_v = e_i\}$. Because such an $\mathbf{x}$ is a vector of vectors of $\mathbb{R}^k$, for convenience we denote by $x(i)$ the ith coordinate of a vector $x \in \mathbb{R}^k$.

Let $\mathbf{x}$ be the vector corresponding to an $(s_1,\ldots,s_k)$-cut $\Pi = (V_1,\ldots,V_k)$. Consider an edge uv. If uv is not Π-transversal, then there exists i such that $x_u = x_v = e_i$, so $\sum_{i=1}^k |x_u(i) - x_v(i)| = 0$. If uv is Π-transversal, then there exist distinct integers i and j such that $x_u = e_i$ and $x_v = e_j$, so $\sum_{i=1}^k |x_u(i) - x_v(i)| = 2$. Hence MULTIWAY CUT may be formulated as follows.

$$\text{Minimize} \quad \textstyle\sum_{uv \in E(G)} w(uv)d(uv) \quad \text{subject to :}$$

$$\begin{array}{rcll} d(uv) & = & \frac{1}{2}\sum_{i=1}^k |x_u(i) - x_v(i)| & \\ x_v & \in & \{e_i \mid 1 \leq i \leq k\} & \text{for all } v \in V \setminus \{v_1,\ldots,v_k\} \\ x_{s_i} & = & e_i & \text{for } 1 \leq i \leq k \end{array} \tag{11.17}$$

This is an integer linear programme, because Equation $(\star)$ can be replaced by following linear inequalities.

$$\begin{aligned} d(uv) &= \frac{1}{2}\sum x_{uv}(i) \\ x_{uv}(i) &\geq x_u(i) - x_v(i) \text{ for all } 1 \leq i \leq k \\ x_{uv}(i) &\geq x_v(i) - x_u(i) \text{ for all } 1 \leq i \leq k \end{aligned}$$

Observe that (11.17) is a generalization of (11.13).

Let Δ_k be the set of vectors of $\mathbb{R}^k$ whose coordinates are non-negative and sum to 1. In other words, $\Delta_k = \{x \mid \sum_{i=1}^{k} x(i) = 1 \text{ and } x(i) \geq 0, \text{for all} 1 \leq i \leq k\}$. One can relax (11.17) into the following linear programme.

$$\begin{array}{lll} \text{Minimize} & \sum_{uv \in E(G)} w(uv)d(uv) & \text{subject to :} \\ & & \\ d(uv) & = \frac{1}{2}\sum_{i=1}^{k} |x_u(i) - x_v(i)| & \\ x_v & \in \Delta_k & \text{for all } v \in V \setminus \{v_1, \ldots, v_k\} \\ x_{s_i} & = e_i & \text{for } 1 \leq i \leq k \end{array} \tag{11.18}$$

For any solution $\mathbf{x}$ of $\Delta_k^{V(G)}$, we denote by $S_{(G,w)}(\mathbf{x})$ or simply $S(\mathbf{x})$ the value $\sum_{uv \in E(G)} w(uv)d(uv)$. The idea of the approximation algorithm is to first find an optimal solution $\mathbf{x}$ of (11.18), and then to derive from this solution an $(s_1, \ldots, s_k)$-cut whose weight is at most $(3/2 - 1/k)S(\mathbf{x})$, and so at most $(3/2 - 1/k)$ times the minimum weight of an $(s_1, \ldots, s_k)$-cut.

In order to derive a multiway cut from the solution, we first transform the edge-weighted graph (G, w) and the solution $\mathbf{x}$ into an edge-weighted graph (G^*, w^*) and an admissible solution (G^*, w^*) to its associated (11.18) such that

(i) $S_{(G,w)}(\mathbf{x}) = S_{(G^*,w^*)}(\mathbf{x}^*)$, and

(ii) for all edge $uv \in E(G^*)$, the vectors x_u^* and x_v^* differ in at most 2 coordinates.

It is easy to see that such (G^*, w^*) and $\mathbf{x}^*$ can be obtained by running the following algorithm.

Algorithm 11.2.

1. $(G^*, w^*) := (G, w)$ and $\mathbf{x} := \mathbf{x}^*$.
2. While there is an edge uv such that x_u^* and x_v^* differ in $m > 2$ coordinates, do
   - Subdivide the edge uv, that is replace the edge uv by two edges uw and wv (where w is a new vertex) with weight $w^*(uw) = w^*(vw) = w^*(uv)$.
   - Choose a vector x_w^* which differs to x_u^* in exactly two coordinates and which differ to x_v^* in fewer than m coordinates.

Let us make few observations that can be easily proved.

Observation 11.19. 1) G^* is a subdivision of G and each time we subdivide an edge uv into (u, w, v), we have $d(uv) = d(uw) + d(wv)$.

2) If $\Pi^* = (V_1^*, \ldots, V_k^*)$ is an $(s_1, \ldots, s_k)$-cut in G^*, then the $(s_1, \ldots, s_k)$-cut $\Pi = (V_1, \ldots, V_k)$ in G, defined by $V_i = V_i^* \cap V(G)$ for all $1 \leq i \leq k$ has weight no greater than the one of Π^*: $w(\Pi) \leq w^*(\Pi^*)$.

We now describe a polynomial-time $(3/2-1/k)$-approximation algorithm for MULTIWAY CUT due to Calinescu et al. [5]. It first find an optimal solution of (11.18) and then deduce an admissible solution of the Multiway Cut Problem.

Let us introduce some notation. For all $1 \leq i \leq k$, let E_i be the set of edges uv of G^* such that $x_u^*(i) \neq x_v^*(i)$ and let $W_i^* = \sum_{e \in E_i} w^*(e)d(e)$. Finally, for $\gamma \in [0,1]$ and $1 \leq i \leq 1$, let $B(s_i,\gamma)$ be the set of vertices of G^* such that $x_v^*(i)$ is at least γ. Observe that if $\gamma > 1/2$, then for $i \neq j$ we have $B(s_i,\gamma) \cap B(s_j,\gamma) = \emptyset$.

Algorithm 11.3 (Multiway Cut Approximation)**.**

1. Find an optimal solution $\mathbf{x}$ to (11.18).
2. Run Algorithm 11.2 to obtain (G^*, w^*) and $\mathbf{x}^*$.
3. Renumber so that $W_k^* = \max\{W_i^* \mid 1 \leq i \leq k\}$.
4. Choose γ at random in $[0,1]$ and choose uniformly at random a permutation σ among the two $(1,2,\ldots,k-1,k)$ and $(k-1,k-2,\ldots,1,k)$.
5. For $i = 1$ to $k-1$, $V_{\sigma(i)}^* := B(s_i,\gamma) \setminus \bigcup_{j<i} V_{\sigma(j)}^*$.
6. $V_k^* := V(G^*) \setminus \bigcup_{j<i} V_{\sigma(j)}^*$.
7. Return the cut $\Pi = (V_i,\ldots,V_k)$, where $V_i = V_i^* \cap V(G)$.

This algorithm is randomized but it can be derandomized.

We shall now show that Algorithm 11.3 returns a $(\frac{3}{2}-\frac{1}{k})$-approximated solution to the Multiway Cut Problem, that is $\mathbf{E}(w(\Pi)) \leq (\frac{3}{2}-\frac{1}{k})w(\Pi_{opt})$ with Π_{opt} a minimum-weight $(s_1,\ldots,s_k)$-cut.

Since $S_{(G^*,w^*)}(\mathbf{x}^*) = S_{(G,w)}(\mathbf{x}) \leq w(\Pi_{opt})$ and $w(\Pi) \leq w^*(\Pi^*)$, it is sufficient to prove

$$\mathbf{E}(w^*(\Pi^*)) \leq \left(\frac{3}{2}-\frac{1}{k}\right) S_{(G^*,w^*)}(\mathbf{x}^*) \tag{11.19}$$

Lemma 11.20. *Let e be an edge of G^*.*

(i) If $e \in E(G^*) \setminus E_k$, then $\mathbf{Pr}(e$ *is* Π^**-transversval)* $\leq \frac{3}{2}d(e)$.

(ii) If $e \in E_k$, then $\mathbf{Pr}(e$ *is* Π^**-transversal)* $\leq d(e)$.

Proof. (i) Let $e = uv$ and let i and j be the two coordinates in which x_u^* and x_v^* differ.

Without loss of generality, we may assume that $x_u^*(i) < x_v^*(i)$. Since x_u^* and x_v^* differ in exactly two coordinates and the sum of the coordinates of each of these vectors equals 1, we have $x_v^*(i) - x_u^*(i) = x_u^*(j) - x_v^*(j)$. In particular, $x_u^*(j) > x_v^*(j)$.

Let us define $B = [x_v^*(j), x_u^*(j)]$ and $A = [x_u^*(i), x_v^*(i)]$ if $x_v^*(i) < x_v^*(j)$ and $A = [x_u^*(i), x_v^*(j)]$ otherwise.

It is clear from the definition of $B(s_i, \gamma)$ and Step 5 that we have the following:

(i) If $\gamma \notin A \cup B$, then u and v are in the same part.

(ii) If $\gamma \in A$ and $\sigma(j) < \sigma(i)$, then u and v are in the same part.

But $\mathbf{Pr}(\gamma \in A \text{ and } \sigma(j) < \sigma(i)) = \mathbf{Pr}(\gamma \in A) \times \mathbf{Pr}(\sigma(j) < \sigma(i)) = d(e)/2$ because the two events '$\gamma \in A$' and '$\sigma(j) < \sigma(i)$' are independent. Thus $\mathbf{Pr}(e \in E(\Pi^*)) \leq \mathbf{Pr}(\gamma \in A \cup B) - \mathbf{Pr}(\gamma \in A \text{ and } \sigma(j) < \sigma(i)) \leq \frac{3}{2} d(e)$. □

Finally,

$$\begin{aligned} \mathbf{E}(w^*(\Pi^*)) = \sum_{e \in E(G^*)} w^*(e)\, \mathbf{Pr}(e \in E(\Pi^*)) &\leq \sum_{e \in E_k} w^*(e) d(e) + \frac{3}{2} \sum_{e \in E(G^*) \setminus E_k} w^*(e) d(e) \\ &\leq \left(\frac{3}{2} - \frac{1}{k}\right) \sum_{e \in E(G^*)} w^*(e) d(e) = S_{(G^*, w^*)}(\mathbf{x}^*). \end{aligned}$$

The last inequality holds because $W_k^* = \max\{W_i^* \mid 1 \leq i \leq k\} \geq \frac{2}{k} \sum_{e \in E} w^*(e) d(e)$, since an edge e is in two E_i because the vectors of its two ends differ in two coordinates.

Remark 11.21. Observe that for $k = 2$, Algorithm 11.3 returns an optimal solution. In fact, in that case it makes the determisitic rounding described in Subsection 11.2.1. Indeed, since $k = 2$, two vectors x_u and x_v may only differ in at most 2 coordinates, so Step 2 is useless. Moreover the two permutations of Step 4 are the same. Hence, when $k = 2$, Algorithm 11.3 computes an optimal solution of the fractional relaxation, chooses randomly a real γ in $[0, 1]$ and then apply the rounding as described in Proposition 11.11.

11.4 Unbounded integrality gap

To give some optimal or approximate algorithm, all the rounding techniques works require the integer linear programme to have a small *integrality gap*, that is that the optimal value of the programme and the one of its rational relaxtion are closed to each other. Unfortunately, this is generally not the case.

11.4.1 Hypergraph transversal

Let $H = (V, E)$ be a hypergraph. A *transversal* of H is a set $X \subset V$ if vertices such that for all hyperedge $e \in E$, $e \cap X \neq \emptyset$. The *transversal number* of H, denoted $\tau(H)$ is the minimum cardinality of a transversal of H. A *fractional transversal* of H is a function $f : V \to \mathbb{R}^+$ such that for all hyperedge $e \in E$, $f(e) \geq 1$. The *fractional transversal number* of H is $\tau_f(G) = \min\{f(V) \mid f \text{ fractional transversal of } H\}$. Clearly, for any hypergraph H, $\tau_f(H) \leq \tau(H)$. On the contrary, the transversal number may not be bounded by any function of the fractional

transversal number. Indeed consider the hypergraph $H = (V,E)$ with $|V| = 2k$ and E the set of all k-subsets of V. Then $\tau(H) \geq k+1$ because any set S of k vertices will not intersect the edge $V \setminus S$, and $\tau_f(H) \leq 2$ because the function f, defined by $f(v) = 1/k$ for all $v \in V$, is a fractional transversal.

11.4.2 Graph colouring

Let $G = (V,E)$ be a graph. We denote by $\mathcal{S}(G)$, or simply $\mathcal{S}$, the set of stable sets of G and $\mathcal{S}(G,v)$, or simply $\mathcal{S}(v)$, the set of all those stable sets which include vertex v. Finding the chromatic number $\chi(G)$ of a graph G may then be formulated as an integer linear programme:

$$\begin{array}{lll} \text{Minimize} & \displaystyle\sum_{S\in\mathcal{S}} x_S & \\ \text{Subject to:} & & \\ & \displaystyle\sum_{S\in\mathcal{S}(v)} \geq 1 & \text{for all } v \in V(G) \\ & x_S \in \{0,1\} & \text{for all } S \in \mathcal{S} \end{array} \tag{11.20}$$

The *fractional chromatic number* of G, denoted $\chi_f(G)$, is the optimal value of the fractional relaxation of (11.20). By the Duality Theorem, the fractional chromatic number is equal to the optimal solution of

$$\begin{array}{lll} \text{Maximize} & \displaystyle\sum_{v\in V(G)} y_v & \\ \text{Subject to:} & & \\ & \displaystyle\sum_{v\in S} y_v \leq 1 & \text{for all } S \in \mathcal{S}(G) \\ & y_v \geq 0 & \text{for all } v \in V \end{array} \tag{11.21}$$

The feasible points of this dual linear programme are often called *fractional cliques*, the reason being that every integral feasible point (i.e., $(0,1)$-vector) identifies a clique of the graph considered. This maximization problem is henceforth called the *fractional clique problem*. Despite this pleasing result, it is still NP-hard to compute the fractional chromatic number of a graph: the number of constraints can be exponential in the size of the graph. In fact, it is even difficult to approximate the fractional chromatic number of graphs with n vertices to within a factor of $n^{1/7-\varepsilon}$ for any positive ε [2].

The fractional chromatic number may be a bad estimate for the chromatic number as the gap between those two numbers may be arbitrarily large. An interesting example of graphs for which there is a large gap between the chromatic number and the fractional chromatic number is provided by the family of *Kneser graphs*, which we now define.

For any two positive integers k and n with $k \leq n$, the *Kneser graph* $KG_{n,k}$ is the graph whose vertices are identified with the k-subsets of $\{1,\cdots,n\}$ and such that two vertices are adjacent if and only if the corresponding sets are disjoint. $KG_{n,1}$ is the complete graph K_n, which has chromatic and $KG_{5,2}$ is the Petersen graph, for which $\chi(K_{5,2}) = 3$. The graph $KG_{2k-1,k}$ has no edges, and so chromatic number 1, whereas $KG_{2k,k}$ consists of a perfect matching, and

hence $\chi(KG_{2k,k}) = 2$ for all $k \geq 1$. Note that $KG(n,k)$ is vertex-transitive. Hence its fractional chromatic number is $|V(KG(n,k)|/\alpha(KG(n,k)) = n/k$. See Exercise 11.14.

Proposition 11.22. *For all $k \geq 1$ and $n \geq 2k$, the fractional chromatic number of the Kneser graph $KG_{n,k}$ is $\frac{n}{k}$.*

Theorem 11.23 (Lovász [21])**.** *For all $k \geq 1$ and $n \geq 2k-1$, the chromatic number of the Kneser graph $KG_{n,k}$ is $\chi(KG_{n,k}) = n - 2k + 2$.*

The proof of this theorem is one of the first graph theoretic results obtained by topological means. We refer the reader interested in such methods to the excellent monograph by Matoušek [22]. The short proof that we present next is due to Greene [13]. It uses one of the many versions of the Borsuk-Ulam Theorem, known as the Generalized Lyusternik Shnirel'man Theorem.

Theorem 11.24 (Greene [13])**.** *For any cover $A_1, A_2, \ldots, A_{n+1}$ of the n-dimensional sphere $\mathbb{S}^n$ by $n+1$ sets, each A_i open or closed, there is at least one set A_i containing a pair of antipodal points (i.e. $\{x, -x\} \in A_i$).*

Proof of Theorem 11.23. Set $d := n - 2k + 1$. Let $X \subset \mathbb{S}^d$ be an n-point set such that no hyperplane passing through the center of $\mathbb{S}^d$ contains more than d points of X. This condition is easily met by a set in a suitably general position, since we deal with points in $\mathbb{R}^{d+1}$ and require that no $d+1$ of them lie on a common hyperplane passing through the origin.

Let us suppose that the vertex set of $KG_{n,k}$ is the set $\binom{X}{k}$ of the k-subsets of X (in other words, we identify the elements of $\{1, \ldots, n\}$ with the points of X).

Consider a d-colouring c of $KG_{n,k}$. We shall prove that c is not proper. For $x \in \mathbb{S}^d$, let $H(x)$ be the open hemisphere $H(x)$ centered at x, that is, $H(x) = \{y \in \mathbb{S}^d \mid \langle x, y \rangle > 0\}$. We define sets $A_1, \ldots, A_d \subseteq \mathbb{S}^d$ as follows. A point $x \in \mathbb{S}^d$ is in A_i if and only if there is at least one vertex $F \in \binom{X}{k}$ of colour i contained in $H(x)$. Finally, we set $A_{d+1} = \mathbb{S}^d \setminus (A_1 \cup \cdots \cup A_d)$. Then, A_i is an open set for each $i \in \{1, \cdots, d\}$, while A_{d+1} is closed. By Theorem 11.24, there exists $i \in \{1, \ldots, d+1\}$ and $x \in \mathbb{S}^d$ such that both x and $-x$ belong to A_i.

Suppose first that $i = d+1$. Then $H(x)$ contains at most $k-1$ points of X, and so does $H(-x)$. Therefore, the complement $\mathbb{S}^d \setminus (H(x) \cup H(-x))$, which is an "equator", (that is, the intersection of $\mathbb{S}^d$ with a hyperplane through the origin), contains at least $n - 2k + 2 = d + 1$ points of X. This contradicts the choice of X.

Hence $i \leq d$. So we obtain two disjoint k-tuples coloured i, one in the open hemisphere $H(x)$ and one in the opposite open hemisphere $H(-x)$. Consequently, c is not proper. □

11.5 Exercises

Exercise 11.1. Formulate the Travelling Salesman Problem as an integer linear programme.

Exercise 11.2. Formulate the Hamiltonian Cycle Problem as an integer linear programme.

Exercise 11.3. Show that $\mu_f(C_{2k+1}) = k + 1/2$.

Exercise 11.4. Show that if G is bipartite, then (11.4) has an integral optimal solution.

Exercise 11.5. Let G be a graph and let $\mathbf{A}$ be its incidence matrix. Prove that $\mathbf{A}$ is totally unimodular if and only if G is bipartite.

Exercise 11.6. Prove that the matrix $[I, A'^T]$ of (11.8) is totally unimodular.

Exercise 11.7. Recall that a *stable set* in a graph is a set of pairwise non-adjacent vertices, and that an *edge cover* of a graph G is a set of edges $F \subseteq E(G)$ such that every vertex $v \in V(G)$ is incident to at least one edge of F.

Let G be a graph.
1) Formulate the problems of finding a maximum stable set and finding a minimum edge cover in G as integer linear programmes.
2) Show that the fractional relaxation of these programmes are dual.
3) Deduce the König–Rado Theorem: *In any bipartite graph without isolated vertices, the number of vertices in a maximum stable set is equal to the number of edges in a minimum edge cover.*
4) Waht can you say if G is not bipartite ?

Exercise 11.8. Let D be a digraph.
1) Show that a non-negative circulation in D is a non-negative linear combination of circulation associated with directed cycles.
2) Show that a non-negative integral circulation in D is a non-negative integral linear combination of circulation associated with directed cycles.

Exercise 11.9 (Hoffman's circulation theorem)**.** Let $D = (V, E)$ be a directed digraph and let d and c be two weight function on the arcs of D such that $d(e) \leq c(e)$ for each arc e. Consider the problem of finding a circulation f such that $d(a) \leq f(a) \leq c(a)$ for every arc e.
1) Model this problem as a linear programmme involving the incidence matrix $\mathbf{A}$ of D.
2) Deduce from the total unimodularity of $\mathbf{A}$ Hoffman's circulation theorem: *There exists a circulation f such that $d(e) \leq f(e) \leq c(e)$ for every arc e if and only if for every subset U of V,*

$$\sum_{e \in E(V \setminus U, U)} d(e) \leq \sum_{e \in E(U, V \setminus U)} c(e).$$

Recall that $E(A, B)$ denotes the set of arcs with tail in A and head in B.

Exercise 11.10. Show that the matrix corresponding to (11.14) is not totally unimodular.

Exercise 11.11. Show that the fractional relaxation and a simple rounding yields a 2-approximate algorithm toMAXIMUM MATCHING.

Exercise 11.12. Show that every integer feasible solution $\mathbf{x}$ to the linear programme (11.5) satisfies the inequality

$$\sum_{e \in E(X)} x_e \leq \frac{1}{2}(|X| - 1)$$

for any odd subset X of V of cardinality three or more.
([8] showed that, by adding these inequalities to the set of constraints in (11.5), one obtains a linear programme every optimal solution of which is $(0, 1)$-valued.)

Exercise 11.13. Let $s_1, \ldots, s_k$ be k vertices of an edge-weighted graph (G, w). For all i, consider the graph G_i obtained from G by identifying the s_j, $j \neq i$ in one vertex t_i and let C_i be a minimum-weight (s_i, t_i)-cut in G_i. Assume moreover that $w(C_k) \geq w(C_i)$, for all i.

Show that $\bigcup_{i=1}^{k-1} C_i$ is an $(s_1, \ldots, s_k)$-cut with weight at most $2 - \frac{2}{k}$ times the minimum weight of an $(s_1, \ldots, s_k)$-cut.

Exercise 11.14.

1) Show that for every graph G, $\chi_f(G) \geq \dfrac{|V(G)|}{\alpha(G)}$.

2) Show that if G is vertex-transitive, then $\chi_f(G) = \dfrac{|V(G)|}{\alpha(G)}$.

3) Decuce that $\chi_f(KG_{n,k}) = \frac{n}{k}$.

Bibliography

[1] N. Alon and J. Spencer. *The Probabilistic Method.* Second edition. Wiley-Interscience Series in Discrete Mathematics and Optimization, Wiley, New York, 2000.

[2] S. Arora and C. Lund. Hardness of approximations. In Dorit Hochbaum, editor, *Approximation Algorithms for NP-hard Problems*. PWS Publishing, Boston, 1996.

[3] T. Asano. Approximation algorithms for MAX-SAT: Yannakakis vs. Goemans-Williamson. In *Proceedings of the 3rd Israel Symposium on the Theory of Computing ans Systems*, Ramat Gan, Israel, pp. 24–37, 1997.

[4] S. Bessy and S. Thomassé. Three min-max theorems concerning cyclic orders of strong digraphs. In *Integer Programming and Combinatorial Optimization*, 132–138. Lecture Notes in Comput. Sci., Vol. 3064, Springer, Berlin, 2004.

[5] G. Calinescu, H. Karloff et Y. Rabani. An improved approximation algorithm for multiway cut. *Journal of Computer and System Sciences* 60:564–574, 2000.

[6] E. Dahlhaus, D.S. Johnson, C. H. Papadimitriou, P. D. Seymour et M. Yannakakis. The complexity of multiterminal cuts. *SIAM J. Comput.* 23:864–894, 1994.

[7] I. Dinur and S. Safra. On the hardness of approximating minimum vertex cover. *Annals of Mathematics* 162 (1): 439–485, 2005.

[8] J. Edmonds. Maximum matching and a polyhedron with $0,1$-vertices. *J. Res. Nat. Bur. Standards Sect. B* 69B:125–130, 1965.

[9] E. Egerváry. On combinatorial properties of matrices. *Mat. Lapok.* 38:16–28, Hungarian with German summary, 1931

[10] T. Gallai. Problem 15. In *Theory of Graphs and its Applications* (M. Fiedler, ed.), 161. Czech. Acad. Sci. Publ., 1964.

[11] M. R. Garey and D. S. Johnson, *Computers and intractability. A guide to the theory of NP-completeness.* A Series of Books in the Mathematical Sciences. W. H. Freeman and Co., San Francisco, Calif., 1979.

[12] M. X. Goemans and D. P. Williamson. New 3/4-approximation algorithms for the maximum satisfiability problem. *SIAM Journal on Discrete Mathematics*, 7:656–666, 1994.

[13] J. E. Greene. A new short proof of Kneser's conjecture. *Amer. Math. Monthly*, 109(10):918–920, 2002.

[14] J. Håstad. Some optimal inapproximability results. *Proceedings of the 28th Annaul ACM Symposium on Theory of Computing*, El Paso, Texas, pp. 1–10, 1997.

[15] A. J. Hoffman and J. B. Kruskal. Integral boundary points of convex polyhedra. In *Linear Inequalities and Related Systems* (H. W. Kuhn and A. W. Tucker, eds.), Princeton University Press, pp. 223–246, 1956.

[16] S. Iwata and T. Matsuda. Finding coherent cyclic orders in strong digraphs. *Combinatorica* 28(1):83–88, 2008.

[17] D. Johnson. Approximation algorithms for combinatorial problems. *J. Comput. System Sci.* 9:256–278, 1974.

[18] S. Khot and O. Regev. Vertex cover might be hard to approximate to within $2-\varepsilon$. *Journal of Computer and System Sciences* 74(3): 335–349.

[19] S. Khuller. The Vertex Cover problem. *ACM SIGACT News* 33(2):31–33, 2002.

[20] D. König. Graphs and matrices. *Mat. Fiz. Lapok* 38:116–119, in Hungarian, 1931.

[21] L. Lovász. Kneser's conjecture, chromatic number, and homotopy. *J. Combin. Theory Ser. A*, 25(3):319–324, 1978.

[22] J. Matoušek. *Using the Borsuk-Ulam theorem*. Universitext. Springer-Verlag, Berlin, 2003. Lectures on topological methods in combinatorics and geometry, Written in cooperation with Anders Björner and Günter M. Ziegler.

[23] G. L. Nemhauser and L. E. Trotter. Vertex packings: Structural properties and algorithms. *Math. Program.* 8:232–248, 1975.

[24] H. Poincaré. Second complément à l'analysis situs. *Proceedings of the London Mathematical Society* 32:277–308, 1900.

[25] P. D. Seymour. Decomposition of regular matroids. *Journal of Combinatorial Theory Series B* 28:305–359, 1980.

[26] M. Yannakakis. On the approximation of maximum satisfiability. *Journal of Algorithms* 17:475–502, 1994.