Lecture 6: Approximation via LP Rounding

Let $G = (V, E)$ be an (undirected) graph. A subset $C \subset V$ is called a *vertex cover* for G if for every edge $(v_i, v_j) \in E$ we have $v_i \in C$ or $v_j \in C$ (or both). In other words, for every edge in E at least one of its endpoints is in C.

6.1 Unweighted vertex cover

The unweighted version of the VERTEX COVER problem is to find a vertex cover of minimum size for a given graph G. This problem is NP-complete.

Let's try to come up with an approximation algorithm. A natural greedy approach would be the following. Initialize the cover C as the empty set, and set $E' := E$; the set E' will contain the edges that are not yet covered by C. Now take an edge $(v_i, v_j) \in E'$, put one of its two vertices, say v_i, into C, and remove from E' all edges incident to v_i. Repeat the process until $E' = \emptyset$. Clearly C is a vertex cover after the algorithm has finished. Unfortunately the algorithm has a very bad approximation ratio: there are instances where it can produce a vertex cover of size $|V| - 1$ even though a vertex cover of size 1 exists.

A small change in the algorithm leads to a 2-approximation algorithm. The change is based on the following lower bound. Call two edges $e, e' \in E$ *adjacent* if they share an endpoint.

Lemma 6.1 *Let $G = (V, E)$ be a graph and let* OPT *denote the minimum size of a vertex cover for G. Let $E^* \subset E$ be any subset of edges such that no two edges in E^* are adjacent. Then* $\text{OPT} \geqslant |E^*|$.

Proof. Let C be an optimal vertex cover for G. By definition, any edge $e \in E^*$ must be covered by a vertex in C, and since the edges in E^* are non-adjacent any vertex in C can cover at most one edge in E^*. □

This lemma suggests the following greedy algorithm.

Algorithm *ApproxVertexCover*(V, E)
1. $C \leftarrow \emptyset$; $E' \leftarrow E$
2. ▷ Invariant: C is a vertex cover for the graph $G' = (V, E \setminus E')$
3. **while** $E' \neq \emptyset$
4. **do** Take an arbitrary edge $(v_i, v_j) \in E'$.
5. $C \leftarrow C \cup \{v_i, v_j\}$
6. Remove all edges from E' that are adjacent to (v_i, v_j).
7. **return** C

It is easy to check that the **while**-loop indeed maintains the invariant. After the **while**-loop has terminated—the loop must terminate since at every step we remove at least one edge from E'—we have $E \setminus E' = E \setminus \emptyset = E$. Together with the invariant this implies that the algorithm indeed returns a vertex cover. Next we show that the algorithm gives a 2-approximation.

Theorem 6.2 *Algorithm ApproxVertexCover produces a vertex cover C such that $|C| \leqslant 2 \cdot$* OPT*, where* OPT *is the minimum size of a vertex cover.*

Proof. Let E^* be the set of edges selected in line 4 over the course of the algorithm. Then C consists of the endpoints of the edges in E^*, and so $|C| \leqslant 2|E^*|$. Moreover, no two edges in

E^* are adjacent because as soon as an edge (v_i, v_j) is selected from E' all edges in E' adjacent to (v_i, v_j) are removed from E'. The theorem now follows from Lemma 6.1. □

6.2 Weighted Vertex Cover

Now let's consider a generalization of Vertex Cover, where each vertex $v_i \in V$ has a weight w_i and we want to find a vertex cover of minimum total weight. We call the new problem Weighted Vertex Cover. The first idea that comes to mind to get an approximation algorithm for Weighted Vertex Cover is to generalize *ApproxVertexCover* as follows: instead of selecting an arbitrary edge (v_i, v_j) from E' in line 4, we select the edge of minimum weight (where the weight of an edge is defined as the sum of the weights of its endpoints). Unfortunately this doesn't work: the weight of the resulting cover can be arbitrarily much larger than the weight of an optimal cover. Our new approach will be based on linear programming.

(Integer) linear programming. In a linear-programming problem we are given a linear *cost function* of d real variables $x_1, \ldots, x_d$ and a set of n linear *constraints* on these variables. The goal is to assign values to the variables so that the cost function is minimized (or: maximized) and all constraints are satisfied. In other words, the Linear Programming problem can be stated as follows:

$$\begin{aligned} \text{Minimize} \quad & c_1 x_1 + \cdots + c_d x_d \\ \text{Subject to} \quad & a_{1,1} x_1 + \cdots + a_{1,d} x_d \leqslant b_1 \\ & a_{2,1} x_1 + \cdots + a_{2,d} x_d \leqslant b_2 \\ & \vdots \\ & a_{n,1} x_1 + \cdots + a_{n,d} x_d \leqslant b_n \end{aligned}$$

There are algorithms—for example the so-called *interior-point methods*—that can solve linear programs in time polynomial in the input size.[1] In practice linear programming is often done with with famous *simplex method*, which is exponential in the worst case but works quite well in most practical applications. Hence, if we can formulate a problem as a linear-programming problem then we can solve it efficiently, both in theory and in practice.

There are several problems that can be formulated as a linear-programming problem but with one twist: the variables $x_1, \ldots, x_d$ can not take on real values but only integer values. This is called Integer Linear Programming. (When the variables can only take the values 0 or 1, the problem is called 0/1 Linear Programming.) Unfortunately, Integer Linear Programming and 0/1 Linear Programming are considerably harder than Linear Programming. In fact, Integer Linear Programming and 0/1 Linear Programming are NP-complete. However, formulating a problem as an integer linear program can still be useful, as shall see next.

Approximating via LP relaxation and rounding. Let's go back to Weighted Vertex Cover. Thus we are given a weighted graph $G = (V, E)$ with $V = \{v_1, \ldots, v_n\}$, and we

[1] Here the input size is measured in terms of the number of bits needed to describe the input, so this is different from the usual notion of input size.

want to find a minimum-weight vertex cover. To formulate this as a 0/1 linear program, we introduce a variable x_i for each vertex $v_i \in V$; the idea is to have $x_i = 1$ if v_i is taken into the vertex cover, and $x_1 = 0$ if v_i is not taken into the cover. When is a subset $C \subset V$ is vertex cover? Then C must contain at least one endpoint for every edge (v_i, v_j). This means we must have $x_i = 1$ or $x_j = 1$. We can enforce this by introducing for every edge $(v_i, v_j) \in E$ the constraint $x_i + x_j \geqslant 1$. Finally, we wish to minimize the total weight of the cover, so we get as a cost function $\sum_{i=1}^{n} w_i x_i$. To summarize, solving the weighted vertex-cover problem corresponds to solving the following 0/1 linear-programming problem.

$$\begin{aligned} \text{Minimize} \quad & w_1 x_1 + \cdots + w_n x_n \\ \text{Subject to} \quad & x_i + x_j \geqslant 1 \qquad \text{for all edges } (v_i, v_j) \in E \\ & x_i \in \{0, 1\} \qquad \text{for } 1 \leqslant i \leqslant n \end{aligned} \tag{1}$$

As noted earlier, solving 0/1 linear programs is hard. Therefore we perform *relaxation*: we drop the restriction that the variables can only take integer values and we replace the integrality constraints (1) by

$$0 \leqslant x_i \leqslant 1 \qquad \text{for } 1 \leqslant i \leqslant n \tag{2}$$

This linear program can be solved in polynomial time. But what good is a solution where the variables can take on any real number in the interval $[0, 1]$? A solution with $x_i = 1/3$, for instance, would suggest that we put 1/3 of the vertex v_i into the cover—something that does not make sense. First we note that the solution to our new relaxed linear program provides us with a lower bound.

Lemma 6.3 *Let W denote the value of an optimal solution to the relaxed linear program described above, and let* OPT *denote the minimum weight of a vertex cover. Then* $\text{OPT} \geqslant W$.

Proof. Any vertex cover corresponds to a feasible solution of the linear program, by setting the variables of the vertices in the cover to 1 and the other variables to 0. Hence, the optimal solution of the linear program is at least as good as this solution. (Stated differently: we already argued that an optimal solution of the 0/1-version of the linear program corresponds to an optimal solution of the vertex-cover problem. Relaxing the integrality constraints clearly cannot make the solution worse.) □

The next step is to derive a valid vertex cover—or, equivalently, a feasible solution to the 0/1 linear program—from the optimal solution to the relaxed linear program. We want to do this in such a way that the total weight of the solution does not increase by much. This can simply be done by *rounding*: variables whose value is at least 1/2 are rounded to 1, variables whose value is less than 1/2 are rounded to 0. We thus obtain the following algorithm for Weighted Vertex Cover.

Algorithm *ApproxWeightedVertexCover*(V, E)
1. $\triangleright$ $G = (V, E)$ is a graph where each vertex $v_i \in V$ $(1 \leqslant i \leqslant n)$ has weight w_i.
2. Solve the relaxed linear program corresponding to the given problem:

$$\begin{array}{lll} \text{Minimize} & w_1x_1 + \cdots + w_nx_n & \\ \text{Subject to} & x_i + x_j \geqslant 1 & \text{for all edges } (v_i, v_j) \in E \\ & 0 \leqslant x_i \leqslant 1 & \text{for } 1 \leqslant i \leqslant n \end{array}$$

3. $C \leftarrow \{v_i \in V : x_i \geqslant 1/2\}$
4. **return** C

Theorem 6.4 *Algorithm ApproxWeightedVertexCover is a 2-approximation algorithm.*

Proof. We first argue that the set C returned by the algorithm is a vertex cover. Consider an edge $(v_i, v_j) \in E$. Then $x_i + x_j \geqslant 1$ is one of the constraints of the linear program. Hence, the reported solution to the linear program—note that a solution will be reported, since the program is obviously feasible by setting all variables to 1—has $\max(x_i, x_j) \geqslant 1/2$. It follows that at least one of v_i and v_j will be put into C.

Let $W := \sum_{i=1}^{n} w_i x_i$ be the total weight of the optimal solution to the relaxed linear program. By Lemma 6.5 we have $\textsc{opt} \geqslant W$. Using that $x_i \geqslant 1/2$ for all $v_i \in C$, we can now bound the total weight of C as follows:

$$\sum_{v_i \in C} w_i \;\leqslant\; \sum_{v_i \in C} w_i \cdot 2x_i \;\leqslant\; 2\sum_{v_i \in C} w_i x_i \;\leqslant\; 2\sum_{i=1}^{n} w_i x_i \;=\; 2W \;\leqslant\; 2 \cdot \textsc{opt}$$

□

Note that, as always, the approximation ratio of our algorithm is obtained by comparing the obtained solution to a certain lower bound—in this case the solution to the LP relaxation. The worst-case ratio between the solution to the integer linear program (which models the problem exactly) and its relaxed version is called the *integrality gap*. For approximation algorithms based on rounding the relaxation of an integer linear program, one typically cannot prove a better approximation ratio than the integrality gap.

6.3 Set Cover

Let $Z := \{z_1, \ldots, z_m\}$ be a finite set. A *set cover* for Z is a collection of subsets of Z whose union is Z. The Set Cover problem is, given a set Z and a collection $\mathcal{S} = S_1, \ldots, S_n$ of subsets of Z, to select a minimum number of subsets from $\mathcal{S}$ that together form a set cover for Z.

Set Cover is a generalization of Vertex Cover. This can be seen as follows. Let $G = (V, E)$ be the graph for which we want to obtain a vertex cover. We can construct an instance of Set Cover from G as follows: The set Z is the set of edges of G, and every vertex $v_i \in V$ defines a subset S_i consisting of those edges of which v_i is an endpoint. Then a set cover for the input $Z, S_1, \ldots, S_n$ corresponds to a vertex cover for G. (Note that Set Cover is more general than Vertex Cover, because in the instance of Set Cover defined by a Vertex Cover instance, every element occurs in exactly two sets—in the general problem an element from Z can occur in many subsets.) In Weighted Set Cover every subset S_i has a weight w_i and we want to find a set cover of minimum total weight.

In this section we will develop an approximation algorithm for WEIGHTED SET COVER. To this end we first formulate the problem as a 0/1 linear program: we introduce a variable x_i that indicates whether S_i is in the cover ($x_i = 1$) or not ($x_i = 0$), and we introduce a constraint for each element $z_j \in Z$ that guarantees that z_j will be in at least one of the chosen sets. The constraint for z_j is defined as follows. Let

$$\mathcal{S}(j) := \{i : 1 \leqslant i \leqslant n \text{ and } z_j \in S_i\}.$$

Then one of the chosen sets contains z_j if and only if $\sum_{i \in \mathcal{S}(j)} x_i \geqslant 1$. This leads to the following 0/1 linear program.

$$\begin{aligned} \text{Minimize} \quad & w_1 x_1 + \cdots + w_n x_n \\ \text{Subject to} \quad & \textstyle\sum_{i \in \mathcal{S}(j)} x_i \geqslant 1 \qquad \text{for all } 1 \leqslant j \leqslant m \\ & x_i \in \{0, 1\} \qquad \text{for } 1 \leqslant i \leqslant n \end{aligned} \tag{3}$$

We relax this 0/1 linear program by replacing the integrality constraints in (3) by the following constraints:

$$0 \leqslant x_i \leqslant 1 \qquad \text{for } 1 \leqslant i \leqslant n \tag{4}$$

We obtain a linear program that we can solve in polynomial time. As in the case of WEIGHTED VERTEX COVER, the value of an optimal solution to this linear program is a lower bound on the value of an optimal solution to the 0/1 linear program and, hence, a lower bound on the minimum total weight of a set cover for the given instance:

Lemma 6.5 *Let W denote the value of an optimal solution to the relaxed linear program described above, and let* OPT *denote the minimum weight of a set cover. Then* $\text{OPT} \geqslant W$.

The next step is to use the solution to the linear program to obtain a solution to the 0/1 linear program (or, in other words, to the set cover problem). Rounding in the same way as for the vertex cover problem—rounding variables that are at least 1/2 to 1, and the other variables to 0—does not work: such a rounding scheme will not give a set cover. Instead we use the following *randomized rounding* strategy:

> For each S_i independently, put S_i into the cover C with probability x_i.

Lemma 6.6 *The expected total weight of C is at most* OPT.

Proof. By definition, the total weight of C is the sum of the weights of its subsets. Let's define an indicator random variable Y_i that tells us whether a set S_i is in the cover C:

$$Y_i = \begin{cases} 1 & \text{if } S_i \in C \\ 0 & \text{otherwise} \end{cases}$$

We have

$$\begin{aligned} \text{E}[\text{ weight of } C\text{ }] &= \text{E}[\textstyle\sum_{i=1}^{n} w_i Y_i\text{ }] \\ &= \textstyle\sum_{i=1}^{n} w_i \,\text{E}[Y_i] && \text{(by linearity of expectation)} \\ &= \textstyle\sum_{i=1}^{n} w_i \cdot \Pr[\text{ } S_i \text{ is put into } C\text{ }] \\ &= \textstyle\sum_{i=1}^{n} w_i x_i \\ &\leqslant \text{OPT} && \text{(by Lemma 6.5)} \end{aligned}$$

□

So the total weight of C is very good. Is C is valid set cover? To answer this question, let's look at the probability that an element $z_j \in Z$ is not covered. Recall that $\sum_{i \in \mathcal{S}(j)} x_i \geqslant 1$. Suppose that z_j is present in ℓ subsets, that is, $|\mathcal{S}(j)| = \ell$. To simplify the notation, let's renumber the sets such that $\mathcal{S}(j) = \{1, \ldots, \ell\}$. Then we have

$$\Pr[\ z_j \text{ is not covered }] \ = \ (1 - x_1) \cdot \cdots \cdot (1 - x_\ell) \ \leqslant \ (1 - \frac{1}{\ell})^\ell,$$

where the last inequality follows from the fact that $(1 - x_1) \cdot \cdots \cdot (1 - x_\ell)$ is maximized when the x_i's sum up to exactly 1 and are evenly distributed, that is, when $x_i = 1/\ell$ for all i. Since $(1 - (1/\ell))^\ell \leqslant 1/e$, where $e \approx 2.718$ is the base of the natural logarithm, we conclude that

$$\Pr[\ z_j \text{ is not covered }] \ \leqslant \ \frac{1}{e} \ \approx \ 0.268.$$

So the probability that any element z_j is covered is fairly high. But this is not good enough: there are many elements z_j and even though each one of them has a good chance of being covered, we cannot expect all of them to be covered simultaneously. (This is only to be expected, of course, since Weighted Set Cover is NP-complete, so we shouldn't hope to find an optimal solution in polynomial time.) What we need is that each element z_j is covered *with high probability*. To this end we simply repeat the above procedure t times, for a suitable value of t: we generate covers $C_1, \ldots, C_t$ where each C_s is obtained using the randomized rounding strategy, and we take $C^* := C_1 \cup \cdots \cup C_t$ as our cover. Our final algorithm is thus as follows.

Algorithm *ApproxWeightedSetCover*$(X, \mathcal{S})$
1. $\triangleright$ $X = \{z_1, \ldots, z_m\}$, and $\mathcal{S} = \{S_1, \ldots, S_n\}$, and set $S_i \in \mathcal{S}$ $(1 \leqslant i \leqslant n)$ has weight w_i.
2. Solve the relaxed linear program corresponding to the given problem:

$$\begin{array}{lll} \text{Minimize} & w_1 x_1 + \cdots + w_n x_n & \\ \text{Subject to} & \sum_{i \in \mathcal{S}(j)} x_i \geqslant 1 & \text{for all } 1 \leqslant j \leqslant m \\ & 0 \leqslant x_i \leqslant 1 & \text{for } 1 \leqslant i \leqslant n \end{array}$$

3. $t \leftarrow 2 \ln m$
4. **for** $s \leftarrow 1$ **to** t
5. **do** $\triangleright$ Compute C_s by randomized rounding
6. **for** $i \leftarrow 1$ to n
7. **do** Put S_i into C_s with probability x_i
8. $C^* \leftarrow C_1 \cup \cdots \cup C_t$
9. **return** C^*

Theorem 6.7 *Algorithm ApproxWeightedSetCover computes a collection C^* that is a set cover with probability at least $1 - 1/m$ and whose expected total weight is $O(\text{OPT} \cdot \log m)$.*

Proof. The expected weight of each C_s is at most OPT, so the expected total weight of C^* is at most $t \cdot \text{OPT} = O(\text{OPT} \cdot \log m)$. What is the probability that some fixed element z_j is not covered by any of the covers C_s? Since the covers C_s are generated independently, and each C_s fails to cover z_j with probability at most $1/e$, we have

$$\Pr[\ z_j \text{ is not covered by any } C_s \ ] \leqslant (1/e)^t.$$

Since $t = 2 \ln m$ we conclude that z_j is not covered with probability at most $1/m^2$. Hence,

$$\begin{aligned} \Pr[\text{ all elements } z_j \text{ are covered by } C^* \,] &= 1 - \Pr[\text{ at least one element } z_j \text{ is not covered by } C^* \,] \\ &\leqslant 1 - \textstyle\sum_{j=1}^{m} \Pr[\, z_j \text{ is not covered by } C^* \,] \\ &\leqslant 1 - 1/m \end{aligned}$$

□