Advanced Algorithms Course. Lecture Notes. Part 2

Set Cover

This is a very fundamental problem abstracted from a variety of applications. Given a set U of n elements, and m subsets S_i of U with positive weights w_i, find a set cover with minimal total weight. A **set cover** is a selection from the given sets S_i whose union is still the whole of U.

The Set Cover problem is NP-complete, as it generalizes the Vertex Cover problem. You should be able to give a polynomial-time reduction from Vertex Cover.

A natural greedy rule is to successively add sets S_i to the solution, that cover as many new elements as possible per unit of weight. More formally: Let R denote the set of yet uncovered elements, initially $R := U$. In every step we put some S_i with minimal $w_i/|S_i \cap R|$ in the solution and update R. This is repeated until $R = \emptyset$.

For the following analysis we use the so-called Harmonic sum. It is defined as $H(n) := \sum_{i=1}^{n} 1/i$. Asymptotically it behaves roughly as $\ln n$. (But note that $H(n)$ is quite different from $\ln n$ for small n.)

Let C be the greedy solution, and C^* an optimal set cover, with weight w^*. The natural the algorithm is, deriving a good bound for its approximation ratio is not so trivial. The key idea of the analysis is to "charge" the covered elements as follows.

Let us define $c_s := w_i/|S_i \cap R|$ for each $s \in S_i \cap R$. Intuitively, c_s is the cost paid by the element s for being covered: The total costs w_i for the step are shared between the newly covered elements. The weight of the greedy solution C obviously equals the sum of these costs: $\sum_{s \in U} c_s = \sum_{S_i \in C} w_i$.

Now consider any set $S_k = \{s_1, \ldots, s_d\}$, where the elements of S_k are sorted in the order they are covered by the greedy algorithm. We study how much is paid by the elements of S_k. Just before an element s_j is

covered we have $|S_k \cap R| \geq d - j + 1$, hence $w_k/|S_k \cap R| \leq w_k/(d - j + 1)$. Let S_i be the set that covers this s_j in the greedy algorithm. Since the algorithm always picks an S_i with minimum weight-per-element ratio, this means $w_i/|S_i \cap R| \leq w_k/|S_k \cap R| \leq w_k/(d-j+1)$. Summation of all element costs in S_k now yields $\sum_{s \in S_k} c_s \leq H(|S_k|)w_k$.

Finally, if d denotes the *maximum* size of the sets S_i, the previous inequality becomes $H(d)w_i \geq \sum_{s \in S_i} c_s$ for each i. We also use the trivial inequality $\sum_{S_i \in C^*} \sum_{s \in S_i} c_s \geq \sum_{s \in U} c_s$. Now we can put things together: $H(d)w^* = H(d) \sum_{S_i \in C^*} w_i \geq \sum_{S_i \in C^*} \sum_{s \in S_i} c_s \geq \sum_{s \in U} c_s = \sum_{S_i \in C} w_i$.

This shows that the greedy algorithm has an approximation ratio $H(d) \approx \ln d$. It may be disappointing that the ratio is not constant and grows with d. But it grows only logarithmically, it is constant when the size d is fixed (a frequent case in applications), and ratio $H(d)$ is also the best possible for any polynomial Set Cover algorithm. (The latter fact is very hard to prove. Such hardness-of-approximation results are far beyond the reach of this course. But we mention the fact for your information.)

Weighted Vertex Cover – The Pricing Method

We are given a graph $G = (V, E)$, where we index the nodes by integers, that is, $V = \{1, \ldots, n\}$, and node i has a weight w_i. The problem is to find a vertex cover of minimum weight. (A vertex cover is a subset of nodes that intersects all edges.) This problem is a special case of Weighted Set Cover (why?). Thus we can apply the previous $H(d)$-approximation, where the maximum node degree takes over the role of d. But, luckily, we can obtain a better approximation ratio. It will be the constant 2. This is not only a nice result as such, but also the method we present is of more general relevance in Optimization. Again we use prices, but now already in the algorithm itself, not only in the analysis. The technique is called the pricing method, or primal-dual method, because the given "primal" problem is attacked using some "dual" problem (see below).

Every edge e will pay a price $p_e \geq 0$ for being covered. We will set these prices later. We say that the prices are *fair* if $\sum_{e=(i,j)} p_e \leq w_i$ for all nodes i. That is, the payments of all edges incident to i do not exceed the weight of i. If prices are fair, we clearly have $\sum_{i \in S} \sum_{e=(i,j)} p_e \leq w(S)$ for any subset S of nodes. If S is a vertex cover, every edge appears at least once in this sum, thus $\sum_{e \in E} p_e \leq \sum_{i \in S} \sum_{e=(i,j)} p_e \leq w(S)$. This inequality says that the sum of (any) fair prices is a lower bound on the cost of any vertex cover,

in particular, for the cost of an optimal vertex cover.

Thus, instead of tackling the problem directly, we may construct prices that are fair but as large as possible (this is going to be our "dual" problem) and then construct somehow a cheap vertex cover from these fair prices. In fact, this is easier than one might expect:

We call a node i tight if $\sum_{e=(i,j)} p_e = w_i$. Initially let all $p_e = 0$. Now we take some e without tight endnodes and simply raise p_e until one endnode is tight. This step is repeated as long as possible. After that, let S be the set of tight nodes.

This was the algorithm! We skip a detailed time analysis, but it is easy to see that the algorithm terminates (since every step produces a new tight node) and the time is polynomial.

Clearly S is a vertex cover, otherwise we could do more steps. Moreover, $\sum_{e=(i,j)} p_e = w_i$ for all $i \in S$, by definition of S. Summation over $i \in S$ gives $\sum_{i \in S} \sum_{e=(i,j)} p_e = w(S)$. Every edge e appears at most twice in this sum, hence $w(S) \leq 2 \sum_{e \in E} p_e$. This shows that $w(S)$ has at most twice the weight of an optimal vertex cover.

Final remark: In the unweighted case, when $w_i = 1$ for all nodes i, the edges with positive prices $p_e = 1$ form a maximal (i.e., non-extendible) matching in the graph, and the vertex cover consists of all nodes contained in the edges of this matching.