V. Vazirani, *Approximation algorithms*, Springer, 2003.

2 Set Cover

The set cover problem plays the same role in approximation algorithms that the maximum matching problem played in exact algorithms – as a problem whose study led to the development of fundamental techniques for the entire field. For our purpose this problem is particularly useful, since it offers a very simple setting in which many of the basic algorithm design techniques can be explained with great ease. In this chapter, we will cover two combinatorial techniques: the fundamental greedy technique and the technique of layering. In Part II we will explain both the basic LP-based techniques of rounding and the primal–dual schema using this problem. Because of its generality, the set cover problem has wide applicability, sometimes even in unexpected ways. In this chapter we will illustrate such an application – to the shortest superstring problem (see Chapter 7 for an improved algorithm for the latter problem).

Among the first strategies one tries when designing an algorithm for an optimization problem is some form of the greedy strategy. Even if this strategy does not work for a specific problem, proving this via a counterexample can provide crucial insights into the structure of the problem. Surprisingly enough, the straightforward, simple greedy algorithm for the set cover problem is essentially the best one can hope for for this problem (see Chapter 29 for a formal proof of this statement).

Problem 2.1 (Set cover) Given a universe U of n elements, a collection of subsets of U, $\mathcal{S} = \{S_1, \ldots, S_k\}$, and a cost function $c : \mathcal{S} \to \mathbf{Q}^+$, find a minimum cost subcollection of $\mathcal{S}$ that covers all elements of U.

Define the *frequency* of an element to be the number of sets it is in. A useful parameter is the frequency of the most frequent element. Let us denote this by f. The various approximation algorithms for set cover achieve one of two factors: $O(\log n)$ or f. Clearly, neither dominates the other in all instances. The special case of set cover with $f = 2$ is essentially the vertex cover problem (see Exercise 2.7), for which we gave a factor 2 approximation algorithm in Chapter 1.

2.1 The greedy algorithm

The greedy strategy applies naturally to the set cover problem: iteratively pick the most cost-effective set and remove the covered elements, until all elements are covered. Let C be the set of elements already covered at the beginning of an iteration. During this iteration, define the *cost-effectiveness* of a set S to be the average cost at which it covers new elements, i.e., $c(S)/|S - C|$. Define the *price* of an element to be the average cost at which it is covered. Equivalently, when a set S is picked, we can think of its cost being distributed equally among the new elements covered, to set their prices.

Algorithm 2.2 (Greedy set cover algorithm)

```
1. $C \leftarrow \emptyset$
2. While $C \neq U$ do
       Find the set whose cost-effectiveness is smallest, say $S$.
       Let $\alpha = \frac{c(S)}{|S-C|}$, i.e., the cost-effectiveness of $S$.
       Pick $S$, and for each $e \in S - C$, set $\text{price}(e) = \alpha$.
       $C \leftarrow C \cup S$.
3. Output the picked sets.
```

Number the elements of U in the order in which they were covered by the algorithm, resolving ties arbitrarily. Let $e_1, \ldots, e_n$ be this numbering.

Lemma 2.3 *For each* $k \in \{1, \ldots, n\}$, $\text{price}(e_k) \le \text{OPT}/(n - k + 1)$.

Proof: In any iteration, the leftover sets of the optimal solution can cover the remaining elements at a cost of at most OPT. Therefore, among these sets, there must be one having cost-effectiveness of at most $\text{OPT}/|\overline{C}|$, where $\overline{C} = U - C$. In the iteration in which element e_k was covered, $\overline{C}$ contained at least $n - k + 1$ elements. Since e_k was covered by the most cost-effective set in this iteration, it follows that

$$\text{price}(e_k) \le \frac{\text{OPT}}{|\overline{C}|} \le \frac{\text{OPT}}{n-k+1}.$$

□

From Lemma 2.3 we immediately obtain:

Theorem 2.4 *The greedy algorithm is an H_n factor approximation algorithm for the minimum set cover problem, where* $H_n = 1 + \frac{1}{2} + \cdots + \frac{1}{n}$.

Proof: Since the cost of each set picked is distributed among the new elements covered, the total cost of the set cover picked is equal to $\sum_{k=1}^{n} \text{price}(e_k)$. By Lemma 2.3, this is at most $\left(1 + \frac{1}{2} + \cdots + \frac{1}{n}\right) \cdot \text{OPT}$. □