

Lecture 3: Set Cover

1 Introduction

1.1 The Problem

We are given a set X of skills, and a set P of people, each of which has some skills. We want to make a committee using as few people as possible, such that every skill is possessed by someone in the committee. (In the weighted version of this problem, the goal is to minimize the sum of the salaries of the people.)

1.2 Set Cover Formulation

Here $X = \{x_1, \dots, x_n\}$ is the base set, and $\mathcal{P} = \{P_1, \dots, P_m\}$, $P_i \subseteq X$. Each person is represented as being a subset of the set of skills. Our goal is to find a sub-collection $C \subseteq \mathcal{P}$ of minimum size such that $\cup_{P \in C} P = X$. Note that it is necessary that $\cup_{P \in \mathcal{P}} P = X$, or no solution will exist.

1.3 Hitting Set Formulation

It is also possible to formulate the problem as follows: $P = \{p_1, \dots, p_m\}$ is the base set, and $\mathcal{X} = \{X_1, \dots, X_n\}$, $X_i \subseteq P$. Now each skill X_i is represented by the people who possess it. Our goal in this formulation is to find a subset $H \subseteq P$ of minimal size such that $|H \cap X_i| \neq 0$, for all i . There is a solution only if each skill-set X_i is non-null.

1.4 Equivalence

It only requires simple, polynomial-time algorithms to go between the two formulations stated above. Both problems are NP-Complete. In the sequel we will use the more traditional notation S_i to denote the sets instead of P_i .

We will present a greedy algorithm for the unweighted set cover problem that is an $O(\log n)$ -optimal approximation algorithm. A 1995 result (see Hochbaum, chapter 10) states that this is the best possible approximation factor, unless $P = NP$. The weighted set-cover problem can also be solved using the greedy algorithm, but for variety we instead give an $O(\log n)$ -optimal randomized algorithm using the hitting set formulation.

2 A Simple Greedy Algorithm for the Set Cover Problem

2.1 The Algorithm

At each stage, pick a set S that covers the most of the elements that are still uncovered. Add S to our set cover.

Theorem 2.1.1 *The above Greedy Algorithm produces a solution that is at most $H(\max\{|s| : s \in \mathcal{S}\})$ times optimal, where $H(n) = \sum_{i=1}^n \frac{1}{i} \leq \ln n + 1$.*

Proof

Let C^* denote the size of an optimal set cover. Let S_i denote the i th subset that the greedy algorithm adds.

We can count the number of sets used in the greedy algorithm by counting 1 for each S_i . Instead, we will spread the cost evenly among all the new elements covered for the first time by S_i .

Let c_x denote the cost allocated to x , for each $x \in X$. Note that each node x is charged exactly once, by the first set in which it is contained.

If x is covered for the first time by S_i (i.e. $x \in S_i$ and $x \notin S_j, 1 \leq j < i$) then

$$c_x = \frac{1}{|S_i - (S_1 \cup S_2 \cup \dots \cup S_{i-1})|}$$

The algorithm finds a set C of total cost $|C|$. Of course, C^* also covers X .

$$|C| = \sum_{x \in X} c_x \leq \sum_{S \in C^*} \sum_{x \in S} c_x$$

Lemma 2.1.2 *For all sets S belonging to \mathcal{S} , $\sum_{x \in S} c_x \leq H(|S|)$.*

proof to follow.

If we assume the lemma, the theorem follows:

$$|C| \leq \sum_{S \in C^*} H(|S|) \leq |C^*| H(\max\{|S| : S \in P\})$$

□

Proof of Lemma

Fix $S \in \mathcal{S}$. For all $i = 1, \dots, |C|$, let $u_i = |S - (S_1 \cup S_2 \cup \dots \cup S_i)|$ be the number of elements in S remaining uncovered after S_1, S_2, \dots, S_i have been selected by the algorithm. Define $u_0 = |S|$.

Let k be the least index such that $u_k = 0$, i.e. all elements of S are covered by one of S_1, \dots, S_i . Clearly $u_{i-1} \geq u_i$, and $u_{i-1} - u_i$ elements of S are covered for the first time by S_i , $i = 1, \dots, k$. Then

$$\sum_{x \in S} c_x = \sum_{i=1}^k (u_{i-1} - u_i) \frac{1}{|S_i - (S_1 \cup \dots \cup S_{i-1})|}$$

but we have

$$|S_i - (S_1 \cup \dots \cup S_{i-1})| \geq |S - (S_1 \cup \dots \cup S_{i-1})| = u_{i-1}$$

so

$$\sum_{x \in S} c_x \leq \sum_{i=1}^k \frac{(u_{i-1} - u_i)}{u_{i-1}}$$

We will need an auxillary

Lemma 2.1.3 *if $a < b$ are integers, then $H(b) - H(a) = \sum_{i=a+1}^b \frac{1}{i} \geq \frac{b-a}{b}$.*

proof is left as exercise.

Using this lemma, we have

$$\sum_{x \in S} c_x \leq \sum_{i=1}^k [H(u_{i-1}) - H(u_i)] = H(u_0) - H(u_k)$$

By definition $u_k = 0$, $u_0 = |S|$. Thus

$$\sum_{x \in S} c_x \leq H(|S|)$$

□

3 A Randomized Approximation Algorithm for the Weighted Hitting Set Problem

Let $V = \{v_1, v_2, \dots, v_n\}$, $\mathcal{S} = \{S_1, \dots, S_m\}$, $S_i \subseteq V, \forall i$. For each vertex v_i , there is an associated weight w_i . We want to choose $H \subseteq V$ to minimize $\sum_{v_i \in H} w_i$, and satisfy $\forall S_i, H \cap S_i \neq \emptyset$.

We can formulate this as an integer linear program: for $1 \leq i \leq n$, set

$$x_i = \begin{cases} 1 & \text{if } v_i \in H \\ 0 & \text{otherwise} \end{cases}$$

Then the IP is

$$\begin{aligned} & \text{minimize } \sum_{i=1}^n x_i w_i \\ & \text{subject to: } \sum_{i: v_i \in S_j} x_i \geq 1, \forall S_j, \\ & x_i \in \{0, 1\}, 1 \leq i \leq n \end{aligned}$$

Solving Integer Programs in general is *NP*-hard. We proceed by removing the integrality constraint. The new constraint is:

$$x_i \in [0, 1]$$

The resulting Linear Program is called the “LP relaxation” of the original Integer Program. We can solve this program in polynomial time. This results in a “fractional” solution, where some x_i s are non-integral. Note that the relaxed LP’s solution is at least as good as that of the original LP, as any solution that was feasible in the original LP is also feasible in the relaxed one. We make use of a technique called Randomized Rounding to recover integrality.

Let \hat{x}_i be the value assigned to x_i in an optimal fractional solution. Let $X_i, 1 \leq i \leq n$ be independent random variables such that

$$\Pr[X_i = 1] = \hat{x}_i, \Pr[X_i = 0] = 1 - \hat{x}_i$$

We now compute the expected weight of the resulting cover:

$$E \left[\sum_{i=1}^n w_i X_i \right] = \sum_{i=1}^n w_i E[X_i] = \sum_{i=1}^n w_i \hat{x}_i$$

i.e. the expected size of the solution is the size of the optimal fractional LP. This is less than the size of the optimal Integer LP, which is equal to the sum of the weights of an optimal hitting set.

We need to bound the probability that we get a feasible solution. To answer this question, consider a set S_i . what is the probability that S_i is covered? Say $S_i = u_1, \dots, u_k$ and recall that $\sum_{j=1}^k \hat{x}_j \geq 1$.

$$\begin{aligned} \Pr[S_i \text{ is covered}] &= 1 - \Pr[\text{no } u_j \text{ is covered}] \\ &= 1 - \prod_{j=1}^k (1 - \hat{x}_j) \geq 1 - \left(1 - \frac{1}{k}\right)^k \geq 1 - \frac{1}{e} > \frac{1}{2} \end{aligned}$$

If we repeat the experiment (independently) $\log m$ times, the probability that S_i is not covered in at least one of the experiments is bounded by $1 - \left(\frac{1}{2}\right)^{\log m} = 1 - \frac{1}{2^m}$. Therefore

$$\begin{aligned} &\Pr[\text{any } S \text{ is not covered}] \\ &\leq \sum_{S \in \mathcal{S}} \Pr[S \text{ is not covered}] \\ &< \sum_{S \in \mathcal{S}} \frac{1}{2^m} = m \frac{1}{2^m} = \frac{1}{2} \end{aligned}$$

If we repeat *this* experiment c times, then we get a hitting set with probability $> 1 - \frac{1}{2^c}$. Furthermore, the cost of this cover is $c \log m \text{ OPT} = O(\log m) \text{OPT}$. Note that we can do everything in one round, by setting $\Pr[X_i = 0] = (1 - \hat{x}_i)^{c \log m}$.