15-854: Approximations Algorithms **Lecturer:** Anupam Gupta
Topic: Greedy Algorithms: Set Cover and Edge Disjoint Paths **Date:** 9/21/2005
Scribe: Pall Melsted

4.1 Introduction

In this lecture we continue our discussion on greedy algorithms. We look at the Set Cover problem, Weighted Set Cover and Maximum Edge Disjoint Paths. Although the description of the algorithms is simple, their analysis is not.

4.2 Set Cover

We present a greedy algorithm for the Set Cover problem. The Set Cover problem is as follows: given a universe U with $|U| = n$ and a family of sets $\mathcal{F} = \{S_1, \ldots, S_m\}$ where $S_i \subseteq U$, find the smallest subset of indices $I \subseteq [m]$ such that

$$\bigcup_{i \in I} S_i = U$$

i.e. the sets cover the universe U.

Hardness The Set Cover problem is NP-complete, Lang and Yannakanis[5] showed that it Set Cover has no approximation ratio $c\ln(n)$ for $c < 1/4$ unless $NP \subseteq DTIME(n^{\text{polylog}(n)}$. Feige[3] improved this result and showed that an approximation ratio of $(1 - o(1))\ln(n)$ is not possible unless $NP \subseteq DTIME(n^{\text{polylog}(n)})$.

Algorithm Greedy Set Cover

1. $X \leftarrow U$, $I = \varnothing$
2. Repeat until $X = \varnothing$
   Pick i s.t. S_i covers maximum number of elements in X
   $I \leftarrow I \cup \{i\}$, $X \leftarrow X \setminus S_i$

Theorem 4.2.1 *If OPT has k sets then Greedy Set Cover picks at most $k(\ln(\frac{n}{k}) + 1)$ sets.*

Corollary 4.2.2 *Greedy Set Cover is a $O(\log(n))$ approximation algorithm.*

Corollary 4.2.3 *Suppose $\forall i |S_i| \leq B$, then Greedy Set Cover is a $O(\log(B))$ approximation algorithm.*

Proof of Theorem 4.2.1: For each t let I_t be the indicator set after t rounds. An element x is covered at time t if $x \in \bigcup_{i \in I_t} S_i$ and let n_t be the number of elements covered after t rounds, $n_0 = n$.

Claim 4.2.4 *For $t \geq 1$, $n_t \leq n_{t-1}(1 - \frac{1}{k})$.*

Proof: Look at the sets in OPT before we pick the t-th set, on average they cover $\frac{n_{t-1}}{k}$ elements. So there exists a set in OPT which covers more than $\frac{n_{t-1}}{k}$ of the remaining elements. Since we picked the largest such set, $S_{i'}$ we know it covered more than $\frac{n_{t-1}}{k}$ elements and so

$$n_t = n_{t-1} - |S_{i'}| \leq n_{t-1} - \frac{n_{t-1}}{k} = n_{t-1}(1 - \frac{1}{k})$$

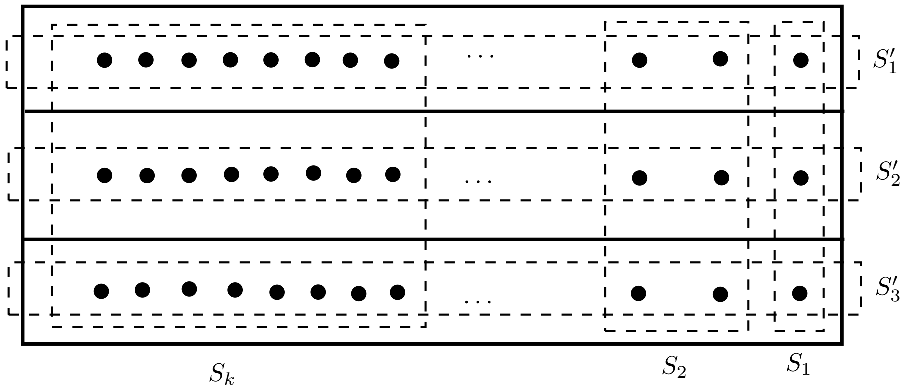
■

Note that $n_0 = n$ and so we have

$$n_t \leq n_0(1 - \frac{1}{k})^t = n(1 - \frac{1}{k})^t \leq ne^{-\frac{t}{k}}$$

set $t = k\ln(\frac{n}{k})$ this will give $n_t \leq n\frac{k}{n} = k$. So after t rounds there are at most k elements left, since each set we pick covers at leas one of the remaining elements we would pick at most k more sets. Thus Greedy will pick at most $n_t + k \leq k\ln(\frac{n}{k}) + k = k(\ln(\frac{n}{k}) + 1)$ sets. ■

Fact 4.2.5 *There are instances where greedy performs badly. Consider figure 4.2.5, with* $n = 3(2^k + \ldots + 2 + 1)$ *elements and sets* $S_1, \ldots, S_k, S'_1, S'_2, S'_3$ *as shown. Clearly Greedy will choose* S_k *since it covers more than half of the elements, then* S_{k-1} *covers half of the remaining elements etc. Note that at each step* S'_1, S'_2 *and* S'_3 *cover only* $\frac{1}{3}$ *of the remaining elements and will never be chosen by Greedy. Thus Greedy chooses* $k = \Omega(\log(n))$ *sets whereas the three sets* S'_1, S'_2, S'_3 *are optimal.*



Weighted Set Cover Weighted Set Cover is a variant of Set Cover where each set has a cost associated with it and we want to pick sets that cover the universe with minimum total cost. Formally, given a universe U with $|U| = n$, a family of sets $\mathcal{F} = S_1, \ldots, S_m$ where set S_i has cost C_i we want

$$\min_{I \subseteq [n]} \sum_{i \in I} C_i \tag{4.2.1}$$

$$\text{s.t} \quad \bigcup_{i \in I} S_i = U \tag{4.2.2}$$

Algorithm Greedy Weighted Set Cover

1. $X \leftarrow U$, $I = \varnothing$

2. Repeat until $X = \varnothing$
   Pick i s.t. $\frac{|X \cap S_i|}{C_i}$ is maximized
   $I \leftarrow I \cup \{i\}$, $X \leftarrow X \setminus S_i$

Theorem 4.2.6 *Weighted Greedy is a* $H_n = 1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n} = O(\log(n))$ *approximation.*

Proof of Theorem 4.2.1: Suppose we've picked t sets, $S_{i_1}, \ldots, S_{i_t}$, and n_t elements remain. If we look at the optimal solution, we see that it spends at most opt to cover the remaining elements. The average cost per element is $\frac{\text{opt}}{n_t}$, so OPT has to contain a set with element cost at most $\frac{\text{opt}}{n_t}$. The set we pick next, $S_{i_{t+1}}$ is the most cost effective set so

$$\frac{c(S_{i_{t+1}})}{|S_{i_{t+1}} \cap X|} \leq \frac{\text{opt}}{n_t}$$

which gives

$$c(S_{i_{t+1}}) \leq |S_{i_{t+1}} \cap X| \frac{\text{opt}}{n_t}$$

Now let $x_t = |S_{i_t}|$, adding all the costs gives

$$\begin{aligned}
\text{cost} &= c(S_{i_1}) + c(S_{i_2}) + \ldots + c(S_{i_k}) \\
&\leq x_1 \frac{\text{opt}}{n} + x_2 \frac{\text{opt}}{n - x_1} + \ldots + x_k \frac{\text{opt}}{n - x_1 - x_2 - \ldots x_{k-1}} \\
&= \text{opt} \left(\underbrace{\frac{1}{n} + \frac{1}{n} + \ldots + \frac{1}{n}}_{x_1 \quad \text{times}} + \underbrace{\frac{1}{n - x_1} + \frac{1}{n - x_1} + \ldots + \frac{1}{n - x_1}}_{x_2 \quad \text{times}} + \ldots \right. \\
&\left. \underbrace{\frac{1}{n - x_1 - \ldots x_{k-1}} + \frac{1}{n - x_1 - \ldots x_{k-1}} + \ldots + \frac{1}{n - x_1 - \ldots x_{k-1}}}_{x_k \quad \text{times}} \right) \\
&\leq \left(\frac{1}{n} + \frac{1}{n-1} + \ldots + \frac{1}{n - x_1 + 1} \right) + \left(\frac{1}{n - x_1} + \ldots + \frac{1}{n - x_1 - x_2 + 1} \right) + \ldots \\
&+ \left(\frac{1}{n - x_1 - \ldots - x_{k-1} + 1} + \ldots + \frac{1}{2} + \frac{1}{1} \right) \\
&= \frac{1}{n} + \frac{1}{n-1} + \ldots + \frac{1}{1} \\
&= H_n
\end{aligned}$$

■

4.3 Edge Disjoint Paths

Maximum Edge Disjoint Paths is an important problem with applications to VLSI design. The problem is NP-complete and Andrews and Zhang[1] showed that there is no $O(\log^{\frac{1}{3}-\epsilon}(|E|))$ approximation algorithm for any $\epsilon > 0$, unless $NP \subseteq ZPTIME(n^{\text{polylog(n)}})$. The algorithm given is due to Kleinberg[4]. Chekuri and Khanna[2] improved this to a $O(\min(n^{\frac{2}{3}}, \sqrt{m}))$ approximation algorithm.

Problem statement Given an undirected graph $G = (V, E)$, a set $T = \{(s_1, t_1), \dots, (s_k, t_k)\}$, s.t. $s_i, t_i \in V$ for all i. Find a maximum set of indices $I \subseteq [n]$ and $\forall i \in I$ a path P_i from s_i to t_i such that $P_i \cap P_j = \varnothing$ for $i, j \in I$ where $i \neq j$ (viewing the path P_i as a set of edges).

Algorithm Greedy EDP

1. Set $G_0 \leftarrow G$ and $I \leftarrow \varnothing$
2. Repeat until all pairs (s_i, t_i), $i \in I$ are disconnected in G_{i-1}
   Find a pair (s_i, t_i), where $i \in [k] \setminus I$ such that the distance between s_i and t_i is minimized in G_{i-1}, choose some shortest path P_i connecting them and set $I \leftarrow I \cup \{i\}$ and $G_i \leftarrow G_{i-1} \setminus P_i$.

Theorem 4.3.1 *Greedy EDP gives an $O(\sqrt{|E|})$ approximation.*

Proof of Theorem 4.3.1: Let $I \subseteq [k]$ be the indices returned by Greedy and $J \subseteq [k]$ be the indices picked by OPT. For each i we have a path P_i and OPT has a path P_j^*. Without loss of generality we can assume we picked indices $1, \dots, |I|$. Let l_i be the length of path P_i, $l_i = |P_i|$. Because the paths were picked as short as possible we have

Fact 4.3.2 $l_1 \leq l_2 \leq \dots \leq l_{|I|}$

We say that a path P is short if $|P| \leq \sqrt{m}$ and long otherwise. Since all paths P_j^* are disjoint and the number of edges is m we have

Fact 4.3.3 *The number of long paths P_j^* in OPT is less than $\sqrt{m}$*

Fact 4.3.4 $|I| \geq 1$

Consider $j \in J \setminus I$, we say that a short path P_j^* is blocked by some P_i if $P_j^* \cap P_i \neq \varnothing$.

Claim 4.3.5 *For each $j \in J \setminus I$, a short path P_j^* is blocked by a short path P_i*

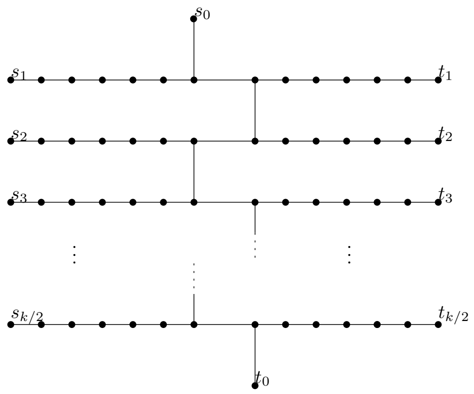
Proof: Take a short path P_j^* where $j \in J \setminus I$, let P_i, $i \in I$ be the shortest path that blocks P_j^*. If $|P_j^*| < |P_i|$ then at the point when P_i was picked, P_j^* was available and shorter and should have been picked by Greedy. A contradiction so $|P_i| \leq |P_j^*| \leq \sqrt{m}$ so P_i is short. ∎

Now let $I_{\text{short}} = \{i \in I : |P_i| \leq \sqrt{m}\}$ and $I_{\text{long}} = I \setminus I_{\text{short}}$ and similarly $J_{\text{short}} = \{j \in J : |P_j^*| \leq \sqrt{m}\}$ and $J_{\text{long}} = J \setminus J_{\text{short}}$. Now for each $i \in I_{\text{short}}$, $|P_i| \leq \sqrt{m}$, so the paths in I_{short} have at most $|I_{\text{short}}|\sqrt{m}$ edges and for each $j \in J_{\text{short}} \setminus I$, P_j^* is blocked by at least one edge and each edge can block at most one P_j^*, since the paths are disjoint. Putting all this together we get the following inequalities.

$$
\begin{aligned}
|J_{\text{short}} \setminus I| &\leq |I_{\text{short}}|\sqrt{m} \leq |I|\sqrt{m} \\
|J_{\text{short}}| &\leq |(J_{\text{short}} \setminus I) \cup I| \leq |I|\sqrt{m} + |I| = (\sqrt{m} + 1)|I| \\
|J_{\text{long}}| &\leq \sqrt{m} \leq |I|\sqrt{m}
\end{aligned}
$$

so $|J| = |J_{\text{short}}| + |J_{\text{long}}| \leq |I|(2\sqrt{m} + 1)$ which shows that Greedy is a $\sqrt{m}$ approximation. ∎

Fact 4.3.6 *Consider figure 4.3.6, where the distance between s_i and t_i is k, for $i = 1, \ldots, \frac{k}{2}$ and the distance between s_0 and t_0 is $k-1$. Greedy EDP will choose (s_0, t_0) as the first pair and remove the only path between s_0 and t_0, disconnecting all other pairs. Clearly the optimal solution is obtained by choosing all pairs (s_i, t_i) for $i = 1, \ldots, \frac{k}{2}$. The ratio between these solutions is $\frac{k}{2} = \Omega(\sqrt{|E|})$.*



References

[1] Matthew Andrews and Lisa Zhang. Hardness of the undirected edge-disjoint paths problem. In *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 276–283, New York, NY, USA, 2005. ACM Press.

[2] Chandra Chekuri and Sanjeev Khanna. Edge disjoint paths revisited. In *SODA '03: Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 628–637, Philadelphia, PA, USA, 2003. Society for Industrial and Applied Mathematics.

[3] Uriel Feige. A threshold of ln n for approximating set cover. *J. ACM*, 45(4):634–652, 1998.

[4] Jon Michael Kleinberg. *Approximation algorithms for disjoint paths problems.* PhD thesis, 1996. Supervisor-Michel X. Goemans.

[5] Carsten Lund and Mihalis Yannakakis. On the hardness of approximating minimization problems. *J. ACM*, 41(5):960–981, 1994.