Network Design Foundation
Fall 2011
Lecture 1

Instructor: Mohammad T. Hajiaghayi
Scribe: Daniel Apon

August 31, 2011

1 Introduction

Network Design Foundation is the theoretical study of "incentive-aware algorithm design" in the context of networks. The algorithmic challenges involved frequently arise from practical scenarios. For example, the Internet is the largest network in the world, and AT&T has millions of dollars riding on the optimization of their Internet-related network design decisions.

While the affiliated topic, *graph algorithms,* focuses on manipulating "dumb" objects toward some goal, *networking algorithms* must deal with "smart," selfish agents. These agents will not necessarily behave how you, the algorithm designer, tell them to. As such, there is a large overlap with *mechanism design and algorithmic game theory,* the general study of algorithms in the context of self-interested agents.

Similarly, an abundance of natural problems encountered in Network Design Foundation are NP-hard to solve exactly. Yet these problems must be solved in practice anyway. As a result, there is also a large overlap with *approximation algorithm design,* the search for efficiently computable solutions that are close to optimal, especially those that appear to work well in practice.

2 Overview

In this lecture, we motivate three classical, NP-complete problems that arise in the context of network design: Set Cover, Unique Coverage, and Maximum Coverage. Then we discuss two approximation algorithms for Set Cover and a matching inapproximability result (up to constant factors).

Scribe: Daniel Apon
Lecture 1 **Date:** 08/31/2011

3 Preliminaries

We are interested in the following three problems:

(1) Set Cover (or Minimum Weighted Set Cover)

Input: A universe $U = \{e_1, \ldots, e_n\}$ of n elements, a collection $\mathcal{S} = \{S_1, \ldots, S_k\}$ of subsets of U such that $U = \bigcup_i S_i$, and a cost function $\mathsf{cost} : \mathcal{S} \to \mathbb{Q}^+$.

Goal: Find a subcollection $\mathcal{S}' \subseteq \mathcal{S}$ such that $\mathcal{S}'$ covers U, i.e. $U = \bigcup_{S \in \mathcal{S}'} S$, with minimum total cost.

(2) Unique Coverage

Input: A universe $U = \{e_1, \ldots, e_n\}$ of n elements and a collection $\mathcal{S} = \{S_1, \ldots, S_k\}$ of subsets of U such that $U = \bigcup_i S_i$.

Goal: Find a subcollection $\mathcal{S}' \subseteq \mathcal{S}$ that maximizes the number of elements covered by exactly one $S \in \mathcal{S}'$.

(3) Maximum Coverage (or Budgeted Maximum Coverage)

Input: A universe $U = \{e_1, \ldots, e_n\}$ of n elements, a collection $\mathcal{S} = \{S_1, \ldots, S_k\}$ of subsets of U such that $U = \bigcup_i S_i$, a cost function $\mathsf{cost} : \mathcal{S} \to \mathbb{Q}^+$, a weight function $\mathit{weight} : U \to \mathbb{Q}^+$, and a constant cost bound $B \in \mathbb{Q}^+$.

Goal: Find a subcollection $\mathcal{S}' \subseteq \mathcal{S}$ that maximizes the total weight of covered elements such that the total cost of the chosen subsets $S \in \mathcal{S}'$ is at most B.

Definition 1 *The* frequency of an element e, *denoted* $f(e)$, *is the number of sets in a collection that contain* e. *Then define the parameter* $F := \max_{e \in U} f(e)$.

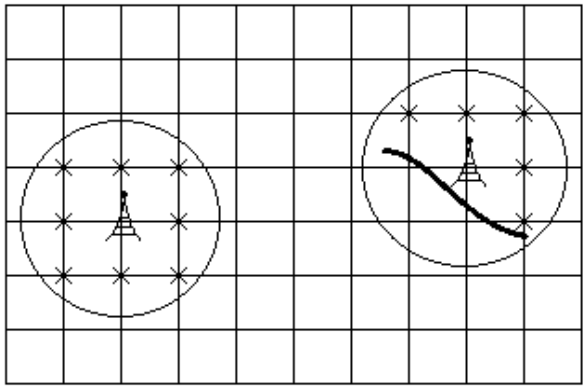
Definition 2 *Let OPT be an optimum solution for a given problem. Then an* α-approximation algorithm *is an algorithm that is guaranteed to output an approximate solution with value/cost* $f(x)$ *that is no worse than a factor* $\alpha \neq 1$ *times the value/cost of OPT on all instances* x. *That is,*

$$\begin{aligned} \mathsf{cost}(\mathsf{OPT}) \leq f(x) \leq \alpha \cdot \mathsf{cost}(\mathsf{OPT}), &\qquad \text{if } \alpha > 1, \\ \alpha \cdot \mathit{value}(\mathsf{OPT}) \leq f(x) \leq \mathit{value}(\mathsf{OPT}), &\qquad \text{if } \alpha < 1. \end{aligned}$$

3.1 Motivation

Wireless networking has many applications for these problems. For example, suppose Verizon wants to build a series of cell towers to cover a new area. Each tower has some coverage radius and a cost associated with constructing or maintaining it. Ideally, we will choose an arrangement that covers the entire area while minimizing the associated cost.

Consider the following example:



In this example, we make the modeling assumption of discretizing the given 2-dimensional space by laying a grid across it. The intersections of grid lines become the legal positions for cell towers as well as the points we must cover (and hence, the elements of the universe $e_i \in U$). The radius of the cell towers determine the set of elements covered, so each possible cell tower location determines a distinct subset of the collection, i.e. $S_j \in \mathcal{S}$. In this example, the size of the subset associated with the right tower is constrained by a geometric obstruction (e.g. a mountain), so its associated elements marked with $\times$'s are fewer.

It is easy to see that Set Cover can be used to find a network of towers that cover every point in the space and has minimum cost. If a limiting budget is added, then Maximum Coverage could be used to find a network that covers the maximum within the budget's constraints. Similarly, Unique Coverage could be used to minimize interference among the transmission towers.

4 Set Cover

We will now show the following:

Theorem 1 *There is a* $\min(\mathcal{O}(\log n), F)$*-approximation algorithm for* Set Cover.

Proof: There are two greedy algorithms, described below, that give an $\mathcal{O}(\log n)$-approximation and F-approximation respectively. We can run both in polynomial time and take the better solution between the two. ■

Remark 1 *In practice, we could decide which to run based on the instance.*

4.1 Greedy Algorithm 1 – $\mathcal{O}(\log n)$-approximation

Consider an instance of Set Cover with universe U and collection $\mathcal{S}$.

Greedy Algorithm 1:

1. set $B \leftarrow \emptyset, C \leftarrow \emptyset, t \leftarrow 1$.
2. while $C \neq U$ (i.e. $U \neq \bigcup_i C_i$), do:
   (a) find a set $B \in \mathcal{S}$ that is the most cost-effective, i.e. the set that minimizes the quantity
   $$\alpha_t := \min_{A \subseteq \mathcal{S}} \frac{\text{cost}(A)}{|A \backslash C|},$$
   (b) select B, and for each $e \in B \backslash C$, set $\text{price}(e) = \alpha_t$
   (c) set $C_t \leftarrow B, t \leftarrow t + 1$.

Output: C (a collection of sets that cover U, i.e. $U = \bigcup_t C_t$)

Analysis:

Suppose we select sets one at a time in an optimal fashion. Let $\text{OPT}_t = \{S_1, \ldots, S_k\}$ be an optimal set cover for the universe $U_t = U \backslash C_t$, where C_t is the collection chosen by **Greedy Algorithm 1** at iteration t.

Lemma 1 *In each iteration* t *of* **Greedy Algorithm 1***, there exists a set with cost-effectiveness at least*
$$\frac{\text{cost}(\text{OPT}_t)}{|U - C|}$$

Proof:
$$\begin{aligned}
\text{cost}(\text{OPT}_t) &= \sum_{i=1}^{k} \text{cost}(S_i) \\
&= \sum_{i=1}^{k} \frac{\text{cost}(S_i)}{|S_i|} |S_i| \\
&\geq \sum_{i=1}^{k} \alpha_t |S_i| \\
&\geq \alpha_t |U - C|.
\end{aligned}$$

Therefore at iteration t,
$$\alpha_t \leq \frac{\text{cost}(\text{OPT}_t)}{|U - C|}$$

■

Let ALG be the solution output by **Greedy Algorithm 1**. Then,

$$\begin{aligned}
\text{cost}(\text{ALG}) &= \sum_{e \in U} \text{price}(e) \\
&= \sum_{i=1}^{n} \text{price}(e_i) \\
&\leq \sum_{i=1}^{n} \frac{\text{cost}(\text{OPT}_i)}{n-i+1} \quad \text{(by definition of } e_i \text{ and Lemma 1)} \\
&= \text{cost}(\text{OPT}) \sum_{i=1}^{n} \frac{1}{i} \\
&= \text{cost}(\text{OPT}) H_n \quad \text{(by definition of the harmonic numbers)} \\
&= \text{cost}(\text{OPT}) \cdot \Theta(\ln(n)).
\end{aligned}$$

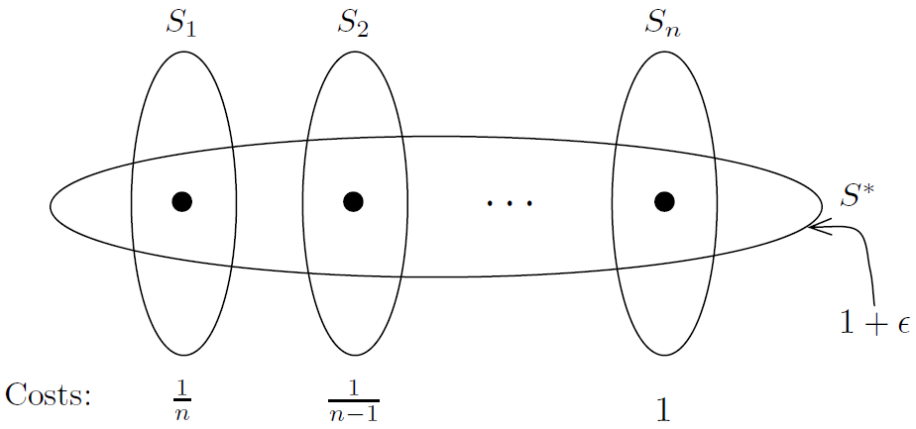
Therefore,

$$\text{cost}(\text{ALG}) \leq \Theta(\ln(n) \cdot \text{cost}(\text{OPT}))$$

and **Greedy Algorithm 1** is an $\mathcal{O}(\log n)$-approximation for Set Cover.

4.2 Worst-case behavior example

In order to see that the above analysis is tight for **Greedy Algorithm 1**, consider the following instance of Set Cover [1], for an arbitrarily small $\epsilon > 0$:



Observe that **Greedy Algorithm 1** will choose sets in the following order – $S_1, S_2, \ldots, S_n$ – despite the optimal solution being to simply choose S^*. The cost of choosing all of the S_i is $1 + \ln(n)$, whereas the cost of the optimal choice of S^* is $1 + \epsilon$.

4.3 Greedy Algorithm 2 – F-approximation

Consider an instance of Set Cover with universe U and collection S where the cost of all sets is 1.

Remark 2 *Note that it is possible to modify the following algorithm to handle the weighted case (i.e. where the cost of all sets is not all 1), though we omit that discussion here.*

Greedy Algorithm 2:

1: Set $A \leftarrow \emptyset$, $U' \leftarrow \emptyset$.

2: While U has an element e not covered by A, add all sets containing e to A and add e to U'.

Output: A (a cover of U)

Analysis:

The number of sets chosen is at most $F \cdot |U'|$ and no set can cover two elements of U'. Therefore,

$$\mathsf{cost}(ALG) \leq |U'| \cdot F$$

and

$$OPT \geq |U'|$$

and **Greedy Algorithm 2** is an F-approximation for Set Cover.

4.4 Almost-Matching Inapproximability Result

Ferge gives a matching inapproximability result for Set Cover (ignoring a constant factor hidden in the $O(\cdot)$ notation):

Theorem 2 (Ferge) *There is no* $(1-\epsilon)\ln n$*-approximation algorithm for* Set Cover *unless* $NP \subseteq DTIME\left(n^{\log\log n}\right)$.

Remark 3 *Note that* $NP \subsetneq DTIME\left(n^{\log\log n}\right)$ *is a mildly stronger assumption than* $P \neq NP$ *as* $n^{\log\log n}$ *is superpolynomial. At the same time, the asymptotic growth of* $n^{\log\log n}$ *is so close to polynomial that the difference is near negligible.*