Tamara Stern 2.9.06

# Set Cover Problem (Chapter 2.1, 12)

*What is the set cover problem?*

Idea: "You must select a minimum number [of any size set] of these sets so that the sets you have picked contain all the elements that are contained in any of the sets in the input (wikipedia)." Additionally, you want to minimize the cost of the sets.

Input:

Ground elements, or Universe $U = \{u_1, u_2, ..., u_n\}$

Subsets $S_1, S_2, ..., S_k \subseteq U$

Costs $c_1, c_2, ..., c_k$

Goal:

Find a set $I \subseteq \{1,2,...,m\}$ that minimizes $\sum_{i \in I} c_i$ , such that $\bigcup_{i \in I} S_i = U$ .

(note: in the un-weighted Set Cover Problem, $c_j = 1$ for all j)

*Why is it useful?*

It was one of Karp's NP-complete problems, shown to be so in 1972.

Other applications: edge covering, vertex cover

Interesting example: IBM finds computer viruses (wikipedia)

elements- 5000 known viruses

sets- 9000 substrings of 20 or more consecutive bytes from viruses, not found in 'good' code

A set cover of 180 was found. It suffices to search for these 180 substrings to verify the existence of known computer viruses.

Another example: Consider General Motors needs to buy a certain amount of varied supplies and there are suppliers that offer various deals for different combinations of materials (Supplier A: 2 tons of steel + 500 tiles for $x; Supplier B: 1 ton of steel + 2000 tiles for $y; etc.). You could use set covering to find the best way to get all the materials while minimizing cost.

*How can we solve it?*

1. Greedy Method – or “brute force” method

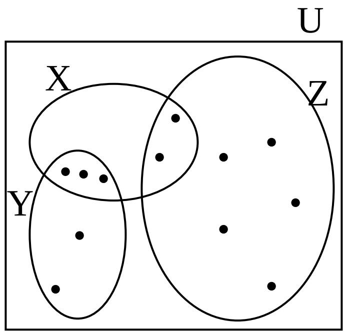
Let C represent the set of elements covered so far

Let cost effectiveness, or $\alpha$, be the average cost per newly covered node

<u>Algorithm</u>

```
1. C ← 0
2. While C ≠ U do
       Find the set whose cost effectiveness is smallest, say S
       Let $\alpha = \frac{c(S)}{|S - C|}$
       For each e ∈ S-C, set price(e) = $\alpha$
       C ← C ∪ S
3. Output picked sets
```

<u>Example</u>

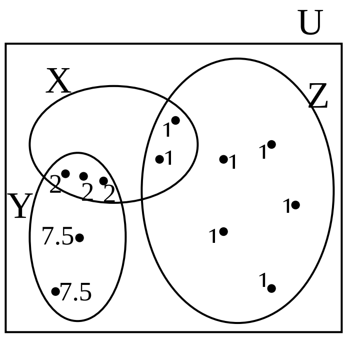


c(X) = 6
c(Y) = 15
c(Z) = 7

Choose Z: $\alpha_Z = \frac{c(Z)}{|S-C|} = \frac{7}{7} = 1$

Choose X: $\alpha_X = \frac{c(X)}{|S-C|} = \frac{6}{3} = 2$

Choose Y: $\alpha_Y = \frac{c(Y)}{|S-C|} = \frac{15}{2} = 7.5$



Total cost = 6 + 15 + 7 = 28

(note: The greedy algorithm is not optimal. An optimal solution would have chosen Y and Z for a cost of 22)

<u>Theorem</u>: The greedy algorithm is an $H_n$ factor approximation algorithm for the minimum set cover problem, where $H_n = 1 + \frac{1}{2} + ... + \frac{1}{n} \approx \log n$.

Proof:

(i) We know $\sum_{e \in U} price(e)$ = cost of the greedy algorithm = $c(S_1) + c(S_2) + ... + c(S_m)$ because of the nature in which we distribute costs of elements.

(ii) We will show $price(e_k) \le \frac{OPT}{n-k+1}$, where $e_k$ is the kth element covered.

Say the optimal sets are $O_1, O_2, ..., O_p$.

So, OPT $\overset{a}{=} c(O_1) + c(O_2) + ... + c(O_p)$.

Now, assume the greedy algorithm has covered the elements in C so far. Then we know the uncovered elements, or $|U - C|$, are at most the intersection of all of the optimal sets intersected with the uncovered elements:

$$|U - C| \overset{b}{\le} |O_1 \cap (U - C)| + |O_2 \cap (U - C)| + ... + |O_p \cap (U - C)|$$

In the greedy algorithm, we select a set with cost effectiveness $\alpha$, where

$$\alpha \overset{c}{\le} \frac{c(O_i)}{|O_i \cap (U - C)|} \quad , \quad i = 1...p.$$

We know this because the greedy algorithm will always choose the set with the smallest cost effectiveness, which will either be smaller than or equal to a set that the optimal algorithm chooses.

Algebra: $c(O_i) \overset{c}{\ge} \alpha \cdot |O_i \cap (U - C)|$

$$\text{OPT} \overset{a}{=} \sum_i c(O_i) \overset{c}{\ge} \alpha \cdot \sum_i |O_i \cap (U - C)| \overset{b}{\ge} \alpha \cdot |U - C|$$

$$\alpha \le \frac{OPT}{|U - C|},$$

Therefore, the price of the k-th element is:

$$\alpha \le \frac{OPT}{n - (k-1)} = \frac{OPT}{n - k + 1}$$

Putting (i) and (ii) together, we get the total cost of the set cover:

$$\sum_{k=1}^{n} price(e_k) \le \sum_{k=1}^{n} \frac{OPT}{n-k+1} = OPT \cdot \left(1 + \frac{1}{2} + ... + \frac{1}{n}\right) = OPT \cdot H_n$$

2. Linear Programming (Chapter 12)

"Linear programming is the problem of optimizing (minimizing or maximizing) a linear function subject to linear inequality constraints (Vazirani 94)."

For example,

$$\begin{array}{lll} \min imize & \sum_{j=1}^{n} c_j x_j & \\ subject\ to & \sum_{j=1}^{n} a_{ij} x_j \geq b_i & i = 1,\ldots,m \\ & x_j \geq 0 & j = 1,\ldots,n \end{array}$$

There exist P-time algorithms for linear programming problems.

Example Problem

$U = \{1,2,3,4,5,6\}$, $S_1 = \{1,2\}$, $S_2 = \{1,3\}$, $S_3 = \{2,3\}$, $S_4 = \{2,4,6\}$, $S_5 = \{3,5,6\}$, $S_6 = \{4,5,6\}$, $S_7 = \{4,5\}$

- variable $x_j$ for set $S_j = \begin{cases} 1\ if\ \ select\ S_j \\ 0\ if\ \ not \end{cases}$

- minimize cost: $Min \sum_{j=1}^{k} c_j x_j = OPT$

- subject to:

$$\begin{array}{ccccccccccccccc} x_1 & + & x_2 & & & & & & & & & & & \geq & 1 \\ x_1 & & & + & x_3 & + & x_4 & & & & & & & \geq & 1 \\ & & x_2 & + & x_3 & & & + & x_5 & & & & & \geq & 1 \\ & & & & & & x_4 & & & + & x_6 & + & x_7 & \geq & 1 \\ & & & & & & & & x_5 & + & x_6 & + & x_7 & \geq & 1 \\ & & & & & & x_4 & + & x_5 & + & x_6 & & & \geq & 1 \end{array}$$

possible solution: $x_1 = x_2 = x_3 = \frac{1}{2}, \quad x_4 = x_5 = x_7 = 0, \quad x_6 = 1 \ \rightarrow \ \sum x_i = 2.5$

This would be a feasible solution – however, we need $x_j \in \{0,1\}$, which is not a valid constraint for linear programming. Instead, we will use $x_j \geq 0$, which will give us all correct solutions, plus more (solutions where $x_j$ is a fraction or greater than 1).

Given a *primal* linear program, we can create a *dual* linear program. Let's revisit linear program:

$$\begin{array}{lll} \min imize & \sum_{j=1}^{n} c_j x_j & \\ subject\ to & \sum_{j=1}^{n} a_{ij} x_j \geq b_i & i = 1,\ldots,m \\ & x_j \geq 0 & j = 1,\ldots,n \end{array}$$

Introducing variables $y_i$ for the ith inequality, we get the dual program:

$$\begin{array}{lll} \max imize & \sum_{i=1}^{m} b_i y_i & \\ subject\ to & \sum_{i=1}^{m} a_{ij} y_j \leq c_i & j = 1,\ldots,n \\ & y_i \geq 0 & i = 1,\ldots,m \end{array}$$

<u>LP-duality Theorem (Vazirani 95)</u>: The primal program has finite optimum iff its dual has finite optimum. Moreover, if $x^* = (x_1^*,\ldots,x_n^*)$ and $y^* = (y_1^*,\ldots,y_m^*)$ are optimal solutions for the primal and dual programs, respectively, then $\sum_{j=1}^{n} c_j x_j^* = \sum_{i=1}^{m} b_i y_i^*$ .

By solving the dual problem, for any feasible solution for $y_1 \rightarrow y_6$, you can guarantee that the optimal result is that solution or greater.

*References:*

http://en.wikipedia.org/wiki/Set_cover_problem.

Vazirani, Vijay. <u>Approximation Algorithms</u>. Chapters 2.1 & 12.

http://www.orie.cornell.edu/~dpw/cornell.ps. p7-18.