

Approximation Algorithms

If we cannot find an **optimal** solution to an NP-hard optimization problem, we might be able to **approximate** it.

Definition 1. *An approximation algorithm has a ratio bound $\rho(n)$, if for any input of size n , the optimal solution $C^*(n)$ and the algorithm solution $C(n)$ satisfy the relation:*

$$\frac{C(n)}{C^*(n)} \leq \rho(n) \text{ for minimization problems}$$

$$\frac{C(n)}{C^*(n)} \geq \rho(n) \text{ for maximization problems.}$$

To establish an approximation ratio, we need to compare the cost of the solution of the algorithm (ALG) to the cost of an optimal solution (OPT).

We have to find “good” polynomial-time computable bounds for OPT .

Vertex Cover

Given a graph $G = (V, E)$, a **vertex cover** of G is a set of vertices $V' \subseteq V$ such that each edge in E is adjacent to at least one vertex in V' .

The **vertex cover optimization problem** is to find a vertex cover of minimum size.

The problem is \mathcal{NP} -hard.

Approximation Algorithm

Find a maximal matching and return all its vertices.

Approximate-Vertex-Cover(G)

1. $C \leftarrow \emptyset$
2. $E' \leftarrow E$
3. While $E' \neq \emptyset$ do
 - (a) Choose an arbitrary edge (u, v) in E'
 - (b) $C \leftarrow \{u, v\}$
 - (c) remove from E' every edge adjacent to u or v
4. return C

Theorem 1. *The algorithm returns a vertex cover, and has a ratio bound of 2.*

Proof.

C is a vertex cover since the algorithm terminates when $E' = \emptyset$.

Let A be the set of edges chosen in line 3.1.

No two edges in A have a common vertex, thus any optimal vertex cover C^* satisfies

$$|C^*| \geq |A|$$

but $|C| = 2|A|$ thus,

$$\frac{|C|}{|C^*|} \leq 2$$

□

Questions

1. Can the approximation guarantee of the above algorithm be improved by a better analysis?

No. We give a **tight example**: an infinite family of instances in which the solution produced by the algorithm is two times OPT.

$K_{n,n}$: complete bipartite graph on $2n$ vertices.

2. Can an approximation algorithm with a better guarantee be designed using the same lower bounding scheme, i.e., size of maximal matching? Assume that we will establish the approx ratio by simply comparing the cost of the solution with the lower bound.

No. Consider K_n (the complete graph on n vertices) with n odd. The size of maximal matching is $(n - 1)/2$ and the size of optimal cover is $n - 1$.

3. Is there an algorithm with a better approximation guarantee?

Set Cover Problem

Given a universe U of n elements and a collection of subsets of U , $\mathcal{S} = \{S_1, S_2, \dots, S_k\}$ and a cost function $c : \mathcal{S} \rightarrow Q^+$, find a **minimum** cost subcollection of \mathcal{S} that covers all elements of U . (Assume that every member of U is contained in some subset.)

Greedy Algorithm:

1. $C = \phi$
2. **while** $C \neq U$ **do**
 Pick a set S such that $c(S)/|S - C|$ is minimized
 $C = C \cup S$
3. Output the picked sets.

Theorem 2. *The greedy algorithm is an H_n factor approximation algorithm for the minimum set cover problem.*

Proof

Distribute the cost of a set picked in an iteration among the *new* elements it covers.

During an iteration, define the *cost-effectiveness* of a set S to be the average cost at which it covers new elements: $c(S)/|S - C|$.

The greedy algo picks the most cost-effective set in the current iteration.

Define the *price* of an element to be the average cost at which it is covered.

Number the elements of U in the order in which they were covered by the algorithm, resolving ties arbitrarily: e_1, e_2, \dots, e_n .

Lemma 1. $price(e_k) \leq \frac{OPT}{n-k+1}, 1 \leq k \leq n.$

Proof. Consider the iteration when e_k was covered.

There must be at least one set having cost-effectiveness at most $\frac{OPT}{|\bar{C}|}$.

Because the leftover sets of an optimal solution can cover the remaining elements at a cost of at most OPT .

Let set S be picked by the greedy algorithm to cover e_k . Then,

$$price(e_k) \leq \frac{OPT}{|\bar{C}|} \leq \frac{OPT}{n-k+1}.$$

since in the iteration when e_k was covered, $|\bar{C}| \geq n - k + 1$. \square

The total cost of the set cover picked by the greedy algorithm is

$$\sum_{k=1}^n price(e_k) = H_n(OPT).$$

Tight Example for the Greedy Algorithm

n singleton sets of cost $1/i$, $i = 1, \dots, n$ each covering a distinct element and a single set of cost $1 + \epsilon$ covering all the elements.

The algorithm outputs a cover of cost H_n while the optimal has a cost of $1 + \epsilon$.

Traveling Salesman Problem

Given a complete graph $G = (V, E)$ with costs $c(u, v)$ on the edges, find a Hamiltonian cycle of minimum cost.

Theorem 3. *The TSP problem (even) with a cost function that satisfies the triangular inequality (Metric-TSP) is NP-complete.*

Approximate-TSP(G, c)

1. Compute a minimum spanning tree T of G .
2. Compute an Euler tour of T starting at an arbitrary vertex a .
3. Compute the TSP by starting at vertex a , by following the Euler tour and short-cutting i.e., skipping vertices that were already visited.

Analysis

Theorem 4. *Approximate-TSP is a 2-approximation algorithm.*

Proof. Let H be the path computed by the algorithm, H^* an optimal path.

For a set of edges X , let $c(X) = \sum_{e \in X} c(e)$.

Since removing an edge from H^* gives a spanning tree

$$c(T) \leq c(H^*).$$

Let W be the Euler tour on T , it visits every edge twice, thus

$$c(W) = 2c(T) \leq 2c(H^*).$$

If W is not an Hamiltonian cycle, we remove vertices from W to get an Hamiltonian cycle.

Assume that W includes the segment $\dots vuw \dots$ and u already appears on the path.

We remove u and connect v directly to w , but

$$c(v, w) \leq c(v, u) + c(u, w)$$

so we don't increase the path cost.

Thus,

$$c(H) \leq c(W) = 2c(T) \leq 2c(H^*).$$

□