Lab09-Approximation Algorithm

CS214-Algorithm and Complexity, Xiaofeng Gao, Spring 2019.

* If there is any problem, please contact TA Jiahao Fan.
* Name: KylinChen Student ID: 517030910155 Email: k1017856853@icloud.com

1. **Metric k-center:** Let $G = (V, E)$ be an complete undirected graph with nonnegative edge costs satisfying the triangle inequality, and k be a positive integer. For any set $S \subseteq V$ and vertex $v \in V$, define $cost(v, S)$ to be the cost of the cheapest edge from v to a vertex in S ($cost(v, S) = 0$ if $v \in S$). The problem is to find a set $S \subseteq V$, with $|S| = k$, so as to minimize $\max_v\{cost(v, S)\}$.

   (a) Design a greedy approximation algorithm (in the form of pseudo code) with approximation ratio 2 for this problem.

   (Basic idea: start with an arbitrary center, and in each round, add the 'farthest' vertex to the center set until there are totaly k centers)

   (b) Prove that your greedy algorithm achieves an approximation ratio of 2 for the metric k-center problem. (Hint: prove by contradiction and use the triangle inequality.)

Solution.

(a) We can get a feasible solution by simple steps below:

- Choose the first center arbitrarily.
- At every step, choose the vertex that is furthest from the current centers to become a center.
- Continue until k centers are chosen.

This algorithm can be implemented by the pseudo code below:

Algorithm 1: *Greedy Algorithm*

```
Input: graph G = (V, E); int k;
Output: set S;
S ← {};
tmp ← V_1;
for i ← 1 to k do
    S ← S ∪ {tmp};
    tmp ← Furthest node from tmp in V/S;
return S;
```

(b) **Theorem:** Greedy algorithm has approximation ratio 2.

Key Observation: Note that the sequence of distances from a new chosen center, to the closest center to it (among previously chosen centers) is non-increasing.

Proof: (1) Once we have chosen k points according to greedy algorithm, we can consider the point M that is furthest from the k chosen centers. If we assume the problem have optimal solution OPT, to prove original theorem, we need to show that the distance from this point M to the closest center is at most $2 \cdot \text{OPT}$.

(2) We can assume that the distance from the furthest point M to all centers is $> 2 \cdot \text{OPT}$, it means

$$cost(M, S_{greedy}) > 2 \cdot OPT$$

This means that distances between all centers are also $> 2 \cdot$ OPT.

(3) According to Key Obeservation, We have $k+1$ points (include k chosen centers) with distances $> 2 \cdot$ OPT between every pair, which means

$$\forall v_1, v_2 \in S_{greedy} \cup \{M\}, v_1 \neq v_2, Distance(v_1, v_2) > 2 \cdot OPT \tag{1}$$

And we set a new notation for this set: $S'_{greedy} = S_{greedy} \cup \{M\}$.

(4) Then we can think about Optimal Solution for this problem. For k chosen points (centers) set S_{opt}, each point has a center of the optimal solution with distance $\leq$ OPT to it.

(5) According to Pigeonhole Principle, there exists a pair of points v_i, v_j in S'_{greedy} with the same nearest center C in the optimal solution set S_{opt}, we have

$$v_1, v_2 \in S'_{greedy}$$

$$cost(v_1, S_{opt}) \leq OPT \Rightarrow Distance(v_1, C) \leq OPT$$

$$cost(v_2, S_{opt}) \leq OPT \Rightarrow Distance(v_2, C) \leq OPT$$

Then refering to triangle inequality, we have

$$Distance(v_1, v_2) < Distance(v_1, C) + Distance(v_2, C) \leq 2 \cdot OPT \tag{2}$$

There exists contradiction between (1) and (2), which proves the original theorem.

2. Let $G = (V, E)$ be a complete undirected graph with nonnegative edge costs satisfying the triangle inequality, and its vertices are partitioned into two sets, R and S. The goal is to find a minimum cost tree in G that contains R and any subset of S. Obviously, a minimum spanning tree (MST) on R is a feasible solution. Prove that finding an MST on R achieves an approximation ratio of 2 for this problem.

Proof.

- (1) We can first assume that an optimal solution S_{opt} is obtained for the problem, and achieve minimum cost OPT. We can use DFS on S_{opt} to prove MST on R achieves an approximation ratio of 2 for this problem.
- (2) All the leaves in the tree S_{opt} must belong to R, Otherwise, one could simply delete the non-R leaves, yielding a feasible solution with less cost.
- (3) refering to an example OPT-tree S_{opt} in Figure.1, we start DFS at an arbitrary R-node, then we can find that all edges have been visited exactly twice(red arrow in Figure.1). Then, it's obvious to find that this cycle can be decomposed into paths between adjacent R-nodes (squares in Figure.1) in the DFS. Fix a pair of such adjacent R-nodes, and consider the shortest path(blue lines in Figure.1) between them. The cost of the shortest path between two adjacent R-node is of course no more than the cost of the path in the optimal tree S_{opt} (triangle inequality).

  We notate the generated cycles(just like blue lines in Figure.1) graph S'(contains all the nodes in R), and notate DFS cycles(just like red arrow(undirect) in Figure.1) graph S_{optDFS}.

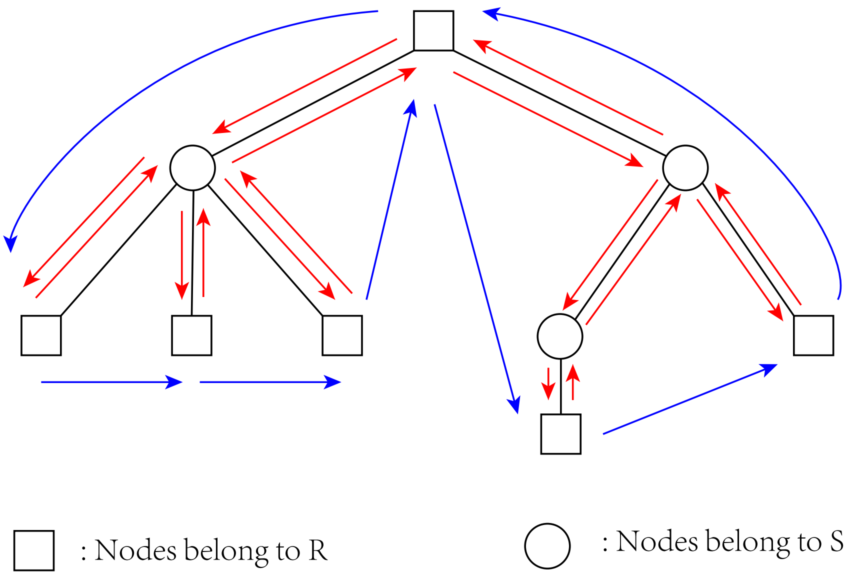

Figure 1: **How to convert OPT Tree to MST on R**

- (4) We know that MST S_{MST} have the minimal weight sum in graph S', so we can get

$$2 \cdot OPT = cost(S_{optDFS}) \geq cost(S') \geq cost(S_{MST})$$

Therefore, MST on R achieves an approximation ratio of 2 for this problem.

3. **Minimum Weighted Vertex Cover:** Consider the weighted version of the Minimum Vertex Cover problem in which a non-negative weight c_i is associated with each vertex v_i and we look for a vertex cover having minimum total weight.

   (a) Given a weighted graph $G = (V, E)$ with a non-negative weight c_i associated with each vertex v_i, please formulate the Minimum Weighted Vertex Cover problem as an integer linear program.

   (b) Prove that the following algorithm finds a feasible solution of the Minimum Weighted Vertex Cover problem with value $m_{LP}(G)$ such that $m_{LP}(G)/m^*(G) \leq 2$.

Algorithm 2: Rounding Weighted Vertex Cover

Input: Graph $G = (V, E)$ with non-negative vertex weights;
Output: Vertex cover V' of G;

Let ILP_{VC} be the integer linear programming formulation of the problem;
Let LP_{VC} be the problem obtained from ILP_{VC} by LP-relaxation;
Let $x^*(G)$ be the optimal solution for LP_{VC};
$V' \leftarrow \{v_i \mid x_i^*(G) \geq 0.5\}$;
return V';

Solution.

(a) First and Foremost, we define a variable x_i for each vertex v_i, which equals 1 ($x_i = 1$) iff v_i is chosen to be included in a vertex cover, otherwise $x_i = 0$.

Then, the minimum weighted vertex cover can be formulated as the following integer linear program:

minimize.

$$\sum_{v=1}^{|V|} c_i x_i$$

subject to.

$$x_i + x_j \geq 1, \forall e = (v_i, v_j) \in E$$
$$x_i \in \{0, 1\}, \forall v_i \in V$$

For section (2), we we can relax the constraint $x_i \in \{0, 1\}$ to $x_i \in [0, 1]$. Futhermore, we can give up the redundant constraint $x_i \leq 1$ to get simplified edition $x_i \geq 0, \forall v_i \in V$. LP approximation can be summed as follow:

minimize.

$$\sum_{v=1}^{|V|} c_i x_i$$

subject to.

$$x_i + x_j \geq 1, \forall e = (v_i, v_j) \in E$$
$$x_i \geq 0, \forall v_i \in V$$

(b) **Proof.**

- (Feasible Solution) For any edge $e = (v_i, v_j) \in E$, by feasibility of x^*, $x_i^* + x_j^* \geq 1$, which means either $x_i^* \geq \frac{1}{2}$ or $x_j^* \geq \frac{1}{2}$. Therefore, at least one of x_i and x_j will be in vertex cover set V'.
- (Approximation Ratio) According to LP, we have

$$m^*(G) = \sum_{v=1}^{|V|} c_i x_i^* \geq \frac{1}{2} \sum_{v_i \in V'} c_i = \frac{1}{2} m_{LP}(G)$$

it obviously prove that $m_{LP}(G)/m^*(G) \leq 2$.

4. Give the corresponding $(I, sol, m, goal)$ for **Metric k-center** and **Minimum Weighted Vertex Cover** respectively.

Solution.

- **Metric k-center**:
  (1) $I = \{(I_1, I_2, k) |\ k \geq 0, \cdots\}$;
  $I_1 = \{G = (V, E) | G\ is\ a\ graph\}$, $I_2 = \{C = (e_i, c_i) |\ e_i \in E, c_i \geq 0, c_i \in triangle\ inequality\}$;
  (2) $sol(G, k) = \{S \subseteq V |\ |S| = k\}$; (feasible solution set and poly-time decidable)
  (3) $m(G, C, S) = \max_v\{cost(v, S)\}$; (poly-time computable function)
  (4) goal = min.
- **Minimum Weighted Vertex Cover**:
  (1) $I = \{(I_1, I_2) | \cdots\}$;
  $I_1 = \{G = (V, E) |\ G\ is\ a\ graph\}$, $I_2 = \{C = (v_i, c_i) |\ v_i \in V, c_i \geq 0\}$;
  (2) $sol(G) = \{S \subseteq V \mid \forall (v_i, v_j) \in E[v_i \in S \vee v_j \in S]\}$; (feasible solution set and poly-time decidable)
  (3) $m(G, C, S) = \sum_{v_i \in S,\ (v_i, c_i)\ exist}(c_i)$; (poly-time computable function)
  (4) goal = min.