Assignment 5

ID: 120037910002 Name: Xingguo Jia Email: jiaxg1998@sjtu.edu.cn

Note: in this assignment, we use m to denote the object function value achieved by the **approximation algorithm** and m^* to denote OPT.

1. Use layering to get a factor f approximation algorithm for set cover, where f is the frequency of the most frequent element. Provide a tight example for this algorithm.

Algorithm: Suppose $I = S_1, S_2, \ldots, S_n$ are the subsets of E, $C \leftarrow \phi, i \leftarrow 0$,$E_0 \leftarrow E$

1. Move all empty sets from I_i to D_i
2. Let $c = min\{w(S)/|S|\}$ for all $S \in I_i$
3. $t_i(S) = c * |S|, w(S) \leftarrow w(S) - t_i(S)$ for all $S \in I_i$
4. $W_i \leftarrow \{S \in I_i \mid w(S) = 0\}, C \leftarrow C \cup W_i$
5. $I_{i+1} \leftarrow I_i - (D_i \cup W_i), E_{i+1} \leftarrow E_i - (D_i \cup W_i)$
6. $i \leftarrow i + 1$, repeat step 2

- We prove that if $w(S)/|S| = c$ for all $S \in I_i$, then $w(I) \leq f * OPT$. We have

$$|E| \leq \sum_{S_i \in U} |S_i| \Rightarrow c|E| \leq c \sum_{S_i \in U} |S_i| = OPT$$

- and

$$\sum_{S_i \in I} |S_i| \leq f * |E| \Rightarrow c * \sum_{S_i \in I} |S_i| \leq c * f * |E| \leq f * OPT$$

- Then we prove this is an f approximation algorithm for set cover. We have

$$w(C) = \sum_{i=0}^{k-1} t_i(C \cap I_i) \leq f * \sum_{i=0}^{k-1} t_i(U \cap I_i) \leq f * w(U) = f * OPT$$

- Tight example:

$$E = \{e_1, e_2, \ldots, e_n\}, i = \{S_1, S_2, \ldots, S_f\}$$

- where

$$S_i = E, w(S_i) = 1, i = 1, \ldots, f$$

- The algorithm would choose all $S_i \in I$, which has weight $m = f$. Optimal solution is $m^* = 1$, Thus

$$\frac{m}{m^*} = f$$

2. Let $G = (V, E)$ be an undirected graph with nonnegative edge costs. S, the senders and R, the receivers, are disjoint subsets of V. The problem is to find a minimum cost subgraph of G such that for every receiver r in R, there is at least one sender s in S such that there is a path connecting r to s in the subgraph. Give a factor 2 approximation algorithm that runs in polynomial time.

Solution:

- Add a vertex v and edges (v, s) for all $s \in S$ with cost 0, and we get graph G'.
- Run MST-based algorithm on G' where $\{v\} \cup R$ is the required set of vertices, and the result is tree T'. We remove vertex v from T' and get T. For any $r \in R$, since there is a path connecting r and v in T and this path must go through some $s \in S$, there is a path connecting r and some $s \in S$.
- Thus, T is a subgraph of G such that for every receiver r in R, there is at least one sender s in S such that there is a path connecting r to s in the T, which is a factor 2 approximation.

3. (Bin Packing) Consider a more restricted algorithm than First-Fit, called Next-Fit, which tries to pack the next item only in the most recently started bin. If it does not fit, it is packed in a new bin. Show that this algorithm also achieves factor 2. Give a factor 2 tight example.

Solution:

- Suppose we use N bins by the Next-Fit algorithm, and each bin has $0 \leq A_i \leq 1$ $(i = 1, \ldots, N)$ amount of item.
- Consider two adjacent bins j and $j+1$ $(1 \leq j \leq N-1)$, we have

$$A_j + A_{j+1} > 1$$

- Thus we have

$$2 * OPT \geq 2 * \sum A_i = A_1 + \sum_{j=1}^{N-1} (A_j + A_{j+1}) + A_N > \sum_{j=1}^{N-1} (A_j + A_{j+1}) > N - 1$$

- Which implies $2 * OPT \geq N$.
- Tight Example: let $0 < \epsilon \leq \frac{1}{n}$, we have $2 * n$ items

$$A_{2i-1} = 2\epsilon, A_{2i} = 1 - \epsilon, i = 1, \ldots, n$$

- Run the Next-Fit algorithm, we need $m = 2n$ bins, but actually, we put $A_{2i-1}, i = 1, \ldots, n$ all in one bin, and each A_{2i} in one bin, we get $m^* = n + 1$ bins. Let $n \to +\infty$, we have

$$\frac{m}{m^*} = 2$$

.

4. (Hamilton cycle) Given an undirected complete graph, each edge is assigned with a nonnegative cost by the function c (eg. the cost for edge (u, v) is $c(u, v)$). Find a Hamilton cycle with the largest cost by the greedy approach, and prove the guarantee factor is 2.

Solution:

Algorithm 1 Greedy-Algorithm for MAX Hamilton cycle

Choose a random vertex $v \in V$, let
$C \leftarrow \{v\}$
while $|C| < |V|$ **do**
  $C \leftarrow C \cup v_m$, s.t. $c(v, v_m) \geq c(v, v_i), \forall v_i \in V$
  $v \leftarrow v_m$
end while
return C

- We prove Algorithm 1 has a guarantee factor of 2. Let

$$C = v_1 v_2 \ldots v_{|V|}$$

- and the optimal solution is

$$O = v_{t_1} v_{t_2} \ldots v_{t_{|V|}}$$

- For $1 \leq i \leq |V| - 1$, we have

$$c(v_{t_i}, v_{t_{i+1}}) \leq max\{c(v_{t_i}, v_{t_i+1}), c(v_{t_{i+1}}, v_{t_{i+1}+1})\} \leq c(v_{t_i}, v_{t_i+1}) + c(v_{t_{i+1}}, v_{t_{i+1}+1})$$

- Thus

$$m = \sum c(v_i, v_{i+1}) = \sum c(v_{t_i}, v_{t_i+1}) =$$

$$\frac{1}{2} * \sum (c(v_{t_i}, v_{t_i+1}) + c(v_{t_{i+1}}, v_{t_{i+1}+1})) \geq \frac{1}{2} * \sum c(v_{t_i}, v_{t_{i+1}}) = \frac{1}{2} * m^* = \frac{1}{2} * OPT$$

5. Given a directed graph $G = (V, E)$, we need to pick a maximum cardinality set of edges from E so that the resulting subgraph is acyclic. Find a factor $\frac{1}{2}$ approximate algorithm for this problem.

Solution:

- Assign a unique integer $f(v_i)$ for each vertex $v_i \in V$. For each edge $(v_i, v_j) \in E$, it is in either of the two sets:

$$A = \{(v_i, v_j) \mid (v_i, v_j) \in E, f(v_i) > f(v_j)\}, B = \{(v_i, v_j) \mid (v_i, v_j) \in E, f(v_i) < f(v_j)\}$$

- There is no cycle in A or B. WLOG, $|A| \geq |B|$. Let A be the result, then we have

$$m = |A| = \frac{|A|}{2} + \frac{|A|}{2} \geq \frac{|A|}{2} + \frac{|B|}{2} = \frac{|E|}{2} = \frac{m^*}{2}$$

6. (Knapsack) Given a set $S = \{a_1, ..., a_n\}$ of objects, with specified non-negative weights and profits, w_i, p_i respectively, and a "knapsack capacity" $B(w_i \leq B)$, find a subset of objects whose total weight is bounded by B and total profit is maximized.

1. Consider two types of greedy algorithms for the knapsack problem. Sort the objects by decreasing **ratio of profit to weight** or only by **profit**, and then greedily pick objects in this order. Show that these two algorithms can be made to perform arbitrarily badly.

2. Combining these two greedy algorithm, pick the more profitable solution in these two algorithms' results. Show that this algorithm achieves an approximation factor of 2.

Solution:

- Let ϵ be an arbitrarily small number, and n be an arbitrarily large number.
- **(ratio of profit to weight)**

$$(w_1, p_1) = (2\epsilon, 3\epsilon), (w_2, p_2) = (B - \epsilon, B - \epsilon)$$

- The algorithm would choose the first item but cannot choose the second item, which has an approximation factor of

$$\frac{m}{m^*} = \frac{3\epsilon}{B - \epsilon} \to 0$$

- **(profit)**

$$(w_1, p_1) = (\epsilon, B - \epsilon), (w_2, p_2) = (\epsilon, B - \epsilon), \ldots, (w_n, p_n) = (\epsilon, B - \epsilon), (w_{n+1}, p_{n+1}) = (B, B)$$

- Then the algorithm would choose the last item, which has an approximation factor of

$$\frac{m}{m^*} = \frac{B}{n * (B - \epsilon)} = \frac{1}{n * (1 - \frac{\epsilon}{B})} \to 0$$

- By combining the two algorithms, we mean running the both two algorithms on the same data, and get the more profitable result. Sort the items by **ratio of profit to weight** and we have k where

$$\sum_{i=1}^{k} w_i \leq B, \sum_{i=1}^{k+1} w_i > B$$

- Since the first $k + 1$ items has the highest ratio of profit to weight and its weight sum is greater than B, then

$$\sum_{i=1}^{k+1} p_i > OPT$$

- Then we have

$$m_{combined} = max\{m_{ratio}, m_{profit}\} \geq \frac{m_{ratio} + m_{profit}}{2}$$
$$= \frac{\sum_{i=1}^{k} p_i + m_{profit}}{2} \geq \frac{\sum_{i=1}^{k} p_i + p_{k+1}}{2} > \frac{OPT}{2}$$

7. (Maximum Cut) Given an undirected graph $G = (V, E)$, the *cardinality maximum cut problem* asks for a partition of V into sets S and $\bar{S}$ so that the number of edges running between these sets is maximized. Find a factor 2 approximation algorithm for this problem.

Solution:

Algorithm 2 Greedy-Algorithm for Maximum Cut

Define $d(S, T) = |\{(u, v) \mid u \in S, v \in T\}|$
$n = |V|$
$S_0 \leftarrow \phi, T_0 \leftarrow \phi$
for $v_i \in V, i = 1, \ldots, n$ **do**
    $S_i \leftarrow S_{i-1}$
    $T_i \leftarrow T_{i-1}$
    if $d(\{v_i\}, S_i) > d(\{v_i\}, T_i)$ **then**
        $T_i \leftarrow T_i \cup \{v_i\}$
    else
        $S_i \leftarrow S_i \cup \{v_i\}$
    end if
end for
return cut S_n, T_n

- We prove Algorithm 2 has approximation factor 2. Since every time we add a new vertex to S_i or T_i, we add

$$max\{d(\{v_i\}, S_i), d(\{v_i\}, T_i)\}$$

- number of edges running between S_n and T_n. Thus we have

$$2m = 2\sum_{i=1}^{n} max\{d(\{v_i\}, S_i), d(\{v_i\}, T_i)\} \geq \sum_{i=1}^{n} (d(\{v_i\}, S_i) + d(\{v_i\}, T_i)) = |E| \geq OPT = m^*$$

$$\Rightarrow \frac{m^*}{m} \leq 2$$

8. Consider the following problem: Given an undirected graph and compute the number of matchings (not the cardinality of a single matching, but the number of different ways of matching) in the graph. Show that if we have an α-approximation algorithm for it for some constant α, then we also have a PTAS.

Solution:

- Based on the α-approximation algorithm f, we design the following algorithm. We have

$$\frac{1}{\alpha} * OPT(G) \leq f(G) \leq \alpha * OPT(G)$$

- First, transform G to $G' = kG$, where G' has k graph Gs with no edges between them. Then run algorithm f on G', we have

$$\frac{1}{\alpha} * OPT(kG) \leq f(kG) \leq \alpha * OPT(kG), \alpha > 1$$

$$\Rightarrow \frac{1}{\alpha} * OPT(G)^k \leq f(kG) \leq \alpha * OPT(G)^k$$

$$\Rightarrow \frac{1}{\sqrt[k]{\alpha}} * OPT(G) \leq \sqrt[k]{f(kG)} \leq \sqrt[k]{\alpha} * OPT(G)$$

- Let $k \to +\infty$, then

$$\frac{1}{\sqrt[k]{\alpha}} \to 1, \sqrt[k]{\alpha} \to 1$$

- and we get a PTAS algorithm $f_1(G) = \sqrt[k]{f(kG)}$.

9. Let G be a complete undirected graph in which all edge lengths are either 1 or 2. Note, that edge lengths satisfy the triangle inequality. Give a factor $4/3$ approximation algorithm for the TSP in this special class of graphs.

Hint: A 2-matching in a graph G is a subset of edges M, such that every vertex is incident to exactly 2 edges of M. A minimum cost 2-matching can be computed in polynomial time.

Solution:

- We have the following algorithm: run minimum cost 2-matching algorithm on G and get minimum cost 2-matching M. Suppose M consists of n cycles. We remove one edge from each cycle and connect them to be one cycle M' and return M'. This runs in polynomial time.
- First, since the minimum size of each cycle in M is 3, we have

$$n \leq \lfloor |V|/3 \rfloor$$

- Then

$$m = C(M') \leq C(M) + n \leq OPT + \lfloor |V|/3 \rfloor \leq OPT + OPT/3 = \frac{4}{3} * OPT$$