CS170–Spring 2019 — Homework 6 Solutions

Ran Liao, SID 3034504227

March 3, 2019

Collaborators:Jingyi Xu, Renee Pu

1 Study Group

Name	SID
Ran Liao	3034504227
Jingyi Xu	3032003885
Renee Pu	3032083302

2 Finding MSTs by Deleting Edges

I'm going to prove every edge deleted by this algorithm is NOT a part of MST. Suppose edge e is going to be deleted and e is a part of MST. Split all vertices into two disjoint parts S_1 and S_2, where edge e cross them. By definition, deleting e will not disconnect the graph. Therefore there exists e' that also crosses these two parts. Since the algorithm processing edges in decreasing weights order, weight of e' must be less than weight of e. Therefore, by cut property, e cannot be a part of MST, which is contradicted to my assumption. Hence, every edge deleted by this algorithm is NOT a part of MST.

3 Huffman Coding

(a) $S(F) = km = m \log n$

(b) Construct a file that n different characters appear equal times.

(c) Construct a file that $n - 1$ character appears only 1 time and the other 1 character appears $m - n - 1$ times. The one character will be encoded with 1 bit length. therefore $H(F) \approx m$ when m is way more larger than n. Therefore, $E(F) = \frac{S(F)}{H(F)} = \frac{m \log n}{m} = O(\log n)$

4 Graph Game

(a) No, since step 2 will never decrease scores.

(b) (i) **Main Idea**

Sort nodes according to their value in decreasing order.

(ii) **Proof of Correctness**

Suppose there's an edge between v and u and $l(v) > l(u)$. Label v before u will always give more scores than the reverse order. The difference should be $l(v) - l(u)$.

(iii) **Runtime Analysis**

Sort will cost $O(|V| \log |V|)$ time. And each edge will be check at most twice in the following process. Therefore, the total runtime is $O(|V| \log |V| + |E|)$.

(c) Use the same example graph showed above, which only has vertex A, B and C. A connects with B, B connects with C. Let $l(A) = 0, l(B) = -1, l(C) = -2$. The optimal solution is label nothing and get 0 score. But my algorithm will get score of -1.

(d) Use the same example again. Let $l(A) = 1, l(B) = -1, l(C) = 1$. The optimal solution is label A, C then B, which will give 2 score. But my friend's algorithm will first delete B and disconnect the graph, which will result in 0 score.

5 Sum of Products

(a) Job i should be assigned to machine $n + 1 - i$.

(b) Since $t_i > t_j$ and $c_{i'} > c_{j'}$, let $t_i = t_j + \Delta t$ and $c_{i'} = c_{j'} + \Delta c$, where $\Delta t, \Delta c > 0$.

Before modification, the cost of assignment is

$$\begin{aligned} \text{cost1} &= t_i c_{i'} + t_j c_{j'} \\ &= (t_j + \Delta t)(c_{j'} + \Delta c) + t_j c_{j'} \\ &= 2t_j c_{j'} + t_j \Delta c + c_{j'} \Delta t + \Delta t \Delta c \end{aligned}$$

After modification, the cost of assignment is

$$\begin{aligned} \text{cost2} &= t_i c_{j'} + t_j c_{i'} \\ &= (t_j + \Delta t) c_{j'} + t_j (c_{j'} + \Delta c) \\ &= 2t_j c_{j'} + t_j \Delta c + c_{j'} \Delta t \end{aligned}$$

Since $\text{cost}_1 - \text{cost}_2 = \Delta t \Delta c > 0$, reassignment will decrease total cost.

(c) Suppose job i is the first job that assigned to machine j, where $j \neq n + 1 - i$. Therefore, all jobs before i are already properly assigned, which implies $j < n + 1 - i$.

So, machine $n+1-i$ can only be assigned with job k, where $k > i$. We can apply modification in part b to these pairs of assignment. Hence, assignment in part a is optimal.

6 Preventing Conflict

(a) **Main Idea**

Process guests one by one. For each guest, put him into the room which he has less enemies in it. i.g. If there're more enemies for him in room A, then put him into room B. Otherwise put him into room A.

(b) **Proof of Correctness**

Denote the number of enemies break by my solution as N. Denote the number of enemies break by optimal solution as O. Due to greedy approach, it's guarantee that more than half of all enemies is break. i.g. $N \geq \frac{1}{2}|E|$. Since $|E| \geq O$, we have $N \geq \frac{1}{2}O$.

(c) **Runtime Analysis**

This algorithm will process each vertex and edge only once. Therefore the runtime is $O(|V| + |E|)$.