CS3230 Tutorial 8 (Greedy Techniques) Sample Solutions

March 25, 2017

1 Question 1

This question is straightforward. Firstly, sort the farmers by their milk prices in ascending order. Purchase the milk with the lowest price first, followed by the second cheapest, ..., until the required N gallons are purchased.

2 Question 2

Sort the jobs in nondecreasing order of their execution times and execute them in that order.

Yes, this greedy algorithm always yields an optimal solution. Let's see how to prove this fact.

Let's prove it by contradiction. We will show that if jobs are executed in some order $i_1, i_2, \ldots, i_n$, in which $t_{i_k} > t_{i_{k+1}}$ for some k, then the total time in the system for such an ordering can be decreased. (Hence, no such ordering can be an optimal solution.) Let us consider the other job ordering, which is obtained by swapping the jobs k and $k+1$. Obviously, the time in the systems will remain the same for all but these two jobs. Therefore, the total time in the stystem for the original ordering before swapping would be:

$$\sum_{j=1}^{n} T_{i_j}$$

where

$$T_{i_k} = \sum_{j=1}^{k} t_{i_j}$$

is the time taken for completing the i_k-the job.

And the total time for the ordering after swapping would be:

$$\sum_{j=1}^{k-1} T_{i_j} + \sum_{j=k+2}^{n} T_{i_j} + (t_{i_1} + \ldots + t_{i_{k-1}} + t_{i_{k+1}}) + (t_{i_1} + \ldots + t_{i_{k-1}} + t_{i_{k+1}} + t_{i_k})$$

The difference in time between the original and the new ordering would be:

$$\sum_{j=1}^{n} T_{i_j} - \left(\sum_{j=1}^{k-1} T_{i_j} + \sum_{j=k+2}^{n} T_{i_j} + (t_{i_1} + \ldots + t_{i_{k-1}} + t_{i_{k+1}}) + (t_{i_1} + \ldots + t_{i_{k-1}} + t_{i_{k+1}} + t_{i_k}) \right)$$

$$= t_{i_k} - t_{i_{k+1}} < 0$$

In other words, the greedy ordering produces a smaller sum and the ordering where we swap the k^{th} and $k+1^{th}$ job is therefore not optimal.

3 Question 3

Tree vertices Priority queue of remaining vertices

Tree vertices	Priority queue of remaining vertices
a(-,-)	b(a,5) c(a,7) d(a,∞) **e(a,2)**
e(a,2)	**b(e,3)** c(e,4) d(e,5)
b(e,3)	**c(e,4)** d(e,5)
c(e,4)	**d(c,4)**
d(c,4)	

The minimum spanning tree found by the algorithm comprises the edges ae, eb, ec, and cd.

4 Question 4

4.a True. Proof: The Prim's algorithm yields a MST solution that contains that smallest edge.

4.b False. An counterexample was given during the tutorial.

4.c True.

Proof: The number of spanning trees for any weighted connected graph is a positive finite number. (At least one spanning tree exists, *e.g.*, the one obtained by a depth-first search traversal of the graph. And the number of spanning trees must be finite because any such tree comprises a subset of edges of the finite set of edges of the given graph.) Hence, one can always find a spanning tree with the smallest total weight among the finite number of the candidates.

Let's prove now that the minimum spanning tree is unique if all the weights are distinct. We will do this by contradiction, *i.e.*, by assuming that there exists a graph $G = (V, E)$ with all distinct weights but with more than one minimum spanning tree. Let $e_1, \ldots, e_{|V|-1}$ be the list of edges composing the minimum spanning tree T_P obtained by Prim's algorithm with some specific vertex as the algorithm's starting point and let T' be another minimum spanning tree. Let $e_i = (v, u)$ be the first edge in the list $e_1, \ldots, e_{|V|-1}$ of the edges of TP which is not in T' (if $T_P \neq T'$, such edge must exist) and let (v, u') be an edge of T' connecting v with a vertex not in the subtree T_{i-1} formed by $e_1, \ldots, e_{i-1}$ (if $i = 1$, T_{i-1} consists of vertex v only). Similarly to the proof of Prim's algorithms correctness, let us replace (v, u') by $e_i = (v, u)$ in T'. It will create another spanning tree, whose weight is smaller than the weight of T' because the weight of $e_i = (v, u)$ is smaller than the weight of (v, u'). (Since e_i was chosen by Prim's algorithm, its weight is the smallest among all the weights on the edges connecting the tree vertices of the subtree T_{i-1} and the vertices adjacent to it. And since all the weights are distinct, the weight of (v, u') must be strictly greater than the weight of $e_i = (v, u)$.) This contradicts the assumption that T' was a minimum spanning tree.

(Note: this is not the only proof. Think of a proof without using Prim's algorithm.)

4.d False. An counterexample was given during the tutorial.

Any bugs and typos, please report to Roger Zimmermann (rogerz@comp.nus.edu.sg).