Problem Set #1

Help

The **due date** for this quiz is **Mon 14 Jul 2014 2:59 PM CST**.

- [ ] **In accordance with the Coursera Honor Code, I (朱顺) certify that the answers here are my own work.**

Question 1

We are given as input a set of n requests (e.g., for the use of an auditorium), with a known start time s_i and finish time t_i for each request i. Assume that all start and finish times are distinct. Two requests *conflict* if they overlap in time --- if one of them starts between the start and finish times of the other. Our goal is to select a maximum-size subset of the given requests that contains no conflicts. (For example, given three requests consuming the intervals $[0, 3]$, $[2, 5]$, and $[4, 7]$, we want to return the first and third requests.) We aim to design a greedy algorithm for this problem with the following form: At each iteration we select a new request i, including it in the solution-so-far and deleting from future consideration all requests that conflict with i. Which of the following greedy rules is guaranteed to always compute an optimal solution?

- ( ) At each iteration, pick the remaining request with the earliest finish time.
- ( ) At each iteration, pick the remaining request with the earliest start time.
- ( ) At each iteration, pick the remaining request with the fewest number of conflicts with other remaining requests (breaking ties arbitrarily).
- ( ) At each iteration, pick the remaining request which requires the least time (i.e., has the smallest value of $t_i - s_i$) (breaking ties arbitrarily).

Question 2

We are given as input a set of n jobs, where job j has a processing time p_j and a deadline d_j. Recall the definition of *completion times* C_j from the video lectures. Given a schedule (i.e., an ordering of the jobs), we define the *lateness* l_j of job j as the amount of time $C_j - d_j$ after its deadline that the job completes, or as 0 if $C_j \leq d_j$. Our goal is to minimize the maximum lateness, $\max_j l_j$. Which of the following greedy rules produces an ordering that minimizes the maximum lateness? You can assume that all processing times and deadlines are distinct.

- Schedule the requests in increasing order of deadline d_j
- Schedule the requests in increasing order of processing time p_j
- Schedule the requests in increasing order of the product $d_j \cdot p_j$
- None of the above.

Question 3

Consider an undirected graph $G = (V, E)$ where every edge $e \in E$ has a given cost c_e. Assume that all edge costs are positive and distinct. Let T be a minimum spanning tree of G and P a shortest path from the vertex s to the vertex t. Now suppose that the cost of every edge e of G is increased by 1 and becomes $c_e + 1$. Call this new graph G'. Which of the following is true about G'?

- T may not be a minimum spanning tree and P may not be a shortest s-t path.
- T must be a minimum spanning tree but P may not be a shortest s-t path.
- T may not be a minimum spanning tree but P is always a shortest s-t path.
- T is always a minimum spanning tree and P is always a shortest s-t path.

Question 4

Suppose T is a minimum spanning tree of the graph G. Let H be an induced subgraph of G. (I.e., H is obtained from G by taking some subset $S \subseteq V$ of vertices, and taking all edges of E that have both endpoints in S.) Which of the following is true about the edges of T that lie in H? You can assume that edge costs are distinct, if you wish.

- They might have non-empty intersection with a minimum spanning tree T_H of H, but at least one of the edges will be missing from T_H

- They are always contained in some minimum spanning tree of H
- They form a minimum spanning tree of H
- They might be disjoint from every minimum spanning tree of H

Question 5

Consider an undirected graph $G = (V, E)$ where edge $e \in E$ has cost c_e. A *minimum bottleneck spanning tree* T is a spanning tree that minimizes the maximum edge cost $\max_{e \in T} c_e$. Which of the following statements is true? Assume that the edge costs are distinct.

- A minimum bottleneck spanning tree is always a minimum spanning tree but a minimum spanning tree is not always a minimum bottleneck spanning tree.
- A minimum bottleneck spanning tree is not always a minimum spanning tree, but a minimum spanning tree is always a minimum bottleneck spanning tree.
- A minimum bottleneck spanning tree is not always a minimum spanning tree and a minimum spanning tree is not always a minimum bottleneck spanning tree.
- A minimum bottleneck spanning tree is always a minimum spanning tree and a minimum spanning tree is always a minimum bottleneck spanning tree.

In accordance with the Coursera Honor Code, I (朱顺) certify that the answers here are my own work.

Submit Answers Save Answers

You cannot submit your work until you agree to the Honor Code. Thanks!