

## Problem Set #1

Quiz, 5 questions

1  
point

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-cardinality 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 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).
- ☐ At each iteration, pick the remaining request with the earliest start time.

1  
point

2.

## Problem Set #1

Quiz, 5 questions

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.

- ☐ None of the other answers are correct.
- ☐ Schedule the requests in increasing order of processing time  $p_j$
- ☐ Schedule the requests in increasing order of the product  $d_j \cdot p_j$
- ☐ Schedule the requests in increasing order of deadline  $d_j$

---

1  
point

3.

## Problem Set #1

Quiz, 5 questions

In this problem you are given as input a graph  $T = (V, E)$  that is a tree (that is,  $T$  is undirected, connected, and acyclic). A *perfect matching* of  $T$  is a subset  $F \subset E$  of edges such that every vertex  $v \in V$  is the endpoint of exactly one edge of  $F$ . Equivalently,  $F$  matches each vertex of  $T$  with exactly one other vertex of  $T$ . For example, a path graph has a perfect matching if and only if it has an even number of vertices.

Consider the following two algorithms that attempt to decide whether or not a given tree has a perfect matching. The *degree* of a vertex in a graph is the number of edges incident to it. (The two algorithms differ only in the choice of  $v$  in line 5.)

Algorithm A:

```

1  While T has at least one vertex:
2    If T has no edges:
3      halt and output "T has no perfect matching."
4    Else:
5      Let v be a vertex of T with maximum degree.
6      Choose an arbitrary edge e incident to v.
7      Delete e and its two endpoints from T.
8  [end of while loop]
9  Halt and output "T has a perfect matching."
```

Algorithm B:

```

1  While T has at least one vertex:
2    If T has no edges:
3      halt and output "T has no perfect matching."
4    Else:
5      Let v be a vertex of T with minimum non-zero degree.
6      Choose an arbitrary edge e incident to v.
7      Delete e and its two endpoints from T.
8  [end of while loop]
9  Halt and output "T has a perfect matching."
```

Is either algorithm correct?

- ☐ Algorithm B always correctly determines whether or not a given tree graph has a perfect matching; algorithm A does not.
- ☐ Neither algorithm always correctly determines whether or not a given tree graph has a perfect matching.
- ☐ Algorithm A always correctly determines whether or not a given tree graph has a perfect matching; algorithm B does not.
- ☐ Both algorithms always correctly determine whether or not a given tree graph has a perfect matching.

1  
point

4.

## Problem Set #1

Quiz, 5 questions

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

1  
point

5.

Suppose  $T$  is a minimum spanning tree of the connected graph  $G$ . Let  $H$  be a connected 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  $S$ . Also, assume  $H$  is connected.) 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. [Choose the strongest true statement.]

- ☐ For every  $G$  and  $H$ , these edges form a spanning tree (but not necessary minimum-cost) of  $H$
- ☐ For every  $G$  and  $H$  and spanning tree  $T_H$  of  $H$ , at least one of these edges is missing from  $T_H$
- ☐ For every  $G$  and  $H$ , these edges form a minimum spanning tree of  $H$
- ☐ For every  $G$  and  $H$ , these edges are contained in some minimum spanning tree of  $H$



I, **David Bai**, understand that submitting work that isn't my own may result in permanent failure of this course or deactivation of my Coursera account.

## Problem Set #1

Quiz, 5 questions

[Learn more about Coursera's Honor Code](#)

Submit Quiz

