CS325: Analysis of Algorithms, Fall 2017

Final Exam

- *I don't know policy:* you may write "I don't know" and *nothing else* to answer a question and receive 25 percent of the total points for that problem whereas a *completely* wrong answer will receive zero.
- There are 8 problems in this exam.

<table>
<thead>
<tr>
<th>Problem 1</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem 2</td>
<td></td>
</tr>
<tr>
<td>Problem 3</td>
<td></td>
</tr>
<tr>
<td>Problem 4</td>
<td></td>
</tr>
<tr>
<td>Problem 5</td>
<td></td>
</tr>
<tr>
<td>Problem 6</td>
<td></td>
</tr>
<tr>
<td>Problem 7</td>
<td></td>
</tr>
<tr>
<td>Problem 8</td>
<td></td>
</tr>
</tbody>
</table>
Problem 1. [3 pts] For each of the following minimum spanning tree algorithms, mark the first five edges that the algorithm would find when it is performed on the given graph.

(a) Borvka:

(b) Jurník, when it starts from the vertex in the center:

(c) Kruskal:

Problem 2. [4 pts] Consider the three minimum spanning tree algorithms (Borvka, Jurník, and Kruskal) as presented in class or in lecture notes. Our proofs of correctness relied on the assumption that the edge weights are distinct.

- Respond True or False for each of the following three statements.

- For each of the False statements, justify your answer by giving an example of a graph with non-distinct edge weights that the algorithm fails to find its MST.

(a) Jurník correctly finds the MST of a graph with non-distinct edge weights. \( \text{True} \)

(b) Kruskal correctly finds the MST of a graph with non-distinct edge weights. \( \text{True} \)

(c) Borvka correctly finds the MST of a graph with non-distinct edge weights. \( \text{False} \)
Problem 3. [3 pts] Let $G = (V, E)$ be a connected graph with $n$ vertices, and $n+1$ edges.

(a) Suppose the weight of every edge of $G$ is 2. What is the weight of the minimum spanning tree of $G$? \[ 2(n-1) \]

(b) Now, assume that $G$ has exactly one edge of weight 1, exactly one edge of weight 4, and the weight of every other edge is 2. What are the possibilities for the weight of the MST of $G$?

Justify your responses to both (a) and (b).

Two poss:

$1 + 2(n-2)$

$1 + 4 + 2(n-3)$

* Any MST selects 1.

Two poss: \{ select 4, not select 4 \}

Problem 4. [4 pts] Which of the following job scheduling algorithms always construct an optimal solution? Mark an algorithm True if it always constructs an optimal solution, and False, otherwise. Provide counterexamples for those that you mark False.

(a) Choose the job $x$ that starts last, discard all jobs that conflict with $x$, and recurse. $\quad T$

(b) Choose the job $x$ with shortest duration, discard all jobs that conflict with $x$, and recurse. $\quad F$

(c) If no jobs conflict, choose them all. Otherwise, discard the job with longest duration and recurse. $\quad F$

(d) Choose a job $x$ that conflicts with the fewest other jobs, discard all jobs that conflict with $x$, and recurse. $\quad F$
Problem 5. [3 pts] What is an optimal Huffman code for the set of frequencies in each of the following cases (specify the codes, or draw the corresponding binary trees)?

(a) \(a : 13, b : 13, c : 13, d : 13, e : 13, f : 13, g : 13, h : 13\).

(b) \(a : 2, b : 2, c : 3, d : 3, e : 5, f : 7\).

Problem 6. [4 pts] For each of the following trees, determine if it can be the optimal Huffman tree for any set of frequencies. For those that can be, find a set of frequencies for which the tree is the optimal Huffman tree, and show the correspondence between the frequencies and the leaves of the tree.

(a) \(\checkmark\)  

(b) \(\checkmark\)  

(c) \(\times\)
Problem 7. [6 pts] For each of the following statements, respond True, False, or Unknown.

(a) If a problem is in $P$ then it is decidable. $\text{T}$

(b) SAT is NP-hard because it can be reduced to CIRCUIT SAT. $\text{F}$

(c) $P \subseteq NP$. $\text{T}$

(d) If a problem is NP-complete then it is in NP. $\text{T}$

(e) If problem $A$ is in NP then it is NP-complete. $\text{F}$

(f) CIRCUIT SAT can be reduced to INDEPENDENT SET in polynomial time. $\text{T}$

(g) There exists a polynomial time algorithm for CIRCUIT SAT. $\text{U}$

(h) If problem $A$ is in NP then there is no polynomial time algorithm to solve $A$. $\text{U}$

Problem 8. [4 pts] A $k$-CNF formula is a conjunction (AND) of a set of clauses, where each clause is a disjunction (OR) of a set of exactly $k$ literals. For example,

$$(a \lor b \lor c \lor \neg d \lor \neg e) \land (\neg a \lor b \lor c \lor \neg x \lor \neg y) \land (\neg x \lor y \lor c \lor d \lor a)$$

is a 5-CNF. The $k$-SAT problem asks if a $k$-CNF formula is satisfiable. In class we saw that 3-SAT is NP-hard. In contrast, 2-SAT is polynomially solvable, as it is mentioned in GA4.

(a) Show that 4-SAT is NP-Complete (For partial credit, specify all the statements you need for showing that 4-SAT is NP-complete, even if you cannot prove them).

(b) Describe a polynomial time algorithm to solve 1-SAT.

(a) (1) 4-SAT is NP-hard:

Reduction from 3-SAT

$$avbc \rightarrow (avbc\neg c\neg x) \land (avbc\neg c\neg x)$$

(2) 4-SAT $\in$ NP

A solution to 4-SAT can be verified in poly time.

(b) A 1-SAT is satisfiable iff it contains no literal and its complement at the same time.