Norwegian University of Science and Technology
Department of Computer and Information Science

Exam authors:
Magnus Lie Hetland
Jon Marius Venstad

Quality control:
Magnar Nedland

Contact during exam:
Magnus Lie Hetland (918 51 949)

Final Exam in
TDT4120 Algorithms and Data Structures

Thursday December 16, 2010
Time: 09:00 – 13:00 Grading date: January 17, 2011

Aids: No printed/handwritten; specific, simple calculator

Language: English

Please read the entire exam before you start, plan your time, and prepare any questions for when the teacher comes to the exam room. Make assumptions where necessary. Keep your answers short and concise. Long explanations that do not directly answer the questions are given little or no weight.

You may describe your algorithms in text, pseudocode, or program code, as long as it is clear how the algorithm works. Short, abstract explanations can be just as good as extensive pseudocode, as long as they are precise enough. Running times are to be given in asymptotic notation, as precisely as possible.

Problem 1

a) (6 %) DIJKSTRA and BELLMAN–FORD solve two different special cases of the same problem. What is the difference between these special cases?

b) (5 %) Which are the requirements on a directed graph if it is to be sorted topologically?

c) (6 %) Is it possible to describe worst-case running time with Ω notation? Explain your reasoning.

d) (5 %) What is linear probing? Please answer briefly.

e) (6 %) What are Hamilton cycles? Why are there no algorithms in the curriculum for finding them?

f) (5 %) A multithreaded computation can be viewed as a computation dag, $G = (V, E)$. Which properties of G are represented by T_1 and T_∞?
(Assume that each strand takes unit time to complete.)

g) (6 %) HEAPSORT is optimal, but RADIX-SORT has a better asymptotic running time. Explain very briefly how this is possible.

h) (6 %) Why isn't it always useful to use memoization in recursive algorithms?

Problem 2

a) (5 %) How will the max-heap $[21, 14, 16, 0, 2, 15, 13]$ look if we insert 15 and then remove the largest element?

b) (6 %) If you are to merge several sorted lists of varying length, and you can only merge two lists at a time (in linear time), how will you minimize the total amount of work needed for all the merge operations? (The final result is to be a single sorted list consisting of the elements from all the original lists.)

c) (5 %) How would you reverse a sequence Q (with length n) using another sequence S in $O(n)$ time if the sequences can only be accessed at their ends? They can be used both as LIFO queues (with PUSH and POP) and as FIFO queues (with ENQUEUE and DEQUEUE). Each of the queue operations take constant time. When you're finished, the elements should be (in reverse order) in Q, and S should be empty.

Problem 3 Imagine that you're facing three decision problems, FOO, BAR, and BAZ. All three problems lie in the class NP. You know that FOO is in NPC and that BAZ is in P. You want to try to construct polynomial reductions between some of the problems in order to reach different conclusions.

Note: The problem is about what you want to do to *try* to reach the various conclusions. If, for example, the problem BAR is *not* in P, it would be impossible to prove that it *is* in P.

a) (3 %) Which problem would you reduce to which in order to show that BAR $\in$ P? Explain briefly.

b) (3 %) Which problem would you reduce to which in order to show that BAR $\in$ NPC? Explain briefly.

c) (3 %) Which problem would you reduce to which in order to show that P = NP? Explain briefly.

Problem 4

a) (5 %) Solve the recurrence $T(n) = 3T(n/2) + n^2$ (asymptotically). Briefly explain your answer.

b) (4 %) Solve the recurrence $T(n) = 16T(n/4) + n$ (asymptotically).

Problem 5

a) (6 %) Let $G = (V, E)$ be a complete, undirected graph, where $V = \{0, \ldots, 9\}$. Let the edge weights of G be given by the following (symmetric) matrix, W:

```
    0  1  2  3  4  5  6  7  8  9
0 [[0, 4, 7, 9, 5, 3, 6, 6, 5, 0],
1  [4, 0, 8, 9, 6, 9, 0, 3, 3, 9],
2  [7, 8, 0, 1, 0, 3, 3, 2, 3, 0],
3  [9, 9, 1, 0, 8, 2, 8, 3, 3, 0],
4  [5, 6, 0, 8, 0, 5, 2, 3, 6, 6],
5  [3, 9, 3, 2, 5, 0, 8, 9, 7, 1],
6  [6, 0, 3, 8, 2, 8, 0, 6, 3, 1],
7  [6, 3, 2, 3, 3, 9, 6, 0, 6, 3],
8  [5, 3, 3, 3, 6, 7, 3, 6, 0, 2],
9  [0, 9, 0, 0, 6, 1, 1, 3, 2, 0]]
```

In other words, $w_{0,3} = 9$, $w_{4,1} = 6$, and $w_{4,2} = 0$, for example. (Edges with a weight of 0 play no special role here; a zero weight does *not* mean that the edge is missing, for example.)

Find the minimum spanning tree T for the graph G, and list the nodes of the path from node 1 to node 4 in the tree T. (As your answer, please write only the path; do *not* describe the full spanning tree.)

Hint: If you work systematically, based on one of the algorithms in the curriculum, this problem will require a lot less work than it might seem to. Choose your algorithm carefully.

b) (5 %) Let $G = (V, E)$ be an unweighted, directed graph with nodes $V = \{1, \ldots, n\}$. Let X be a one-dimensional integer array that can be indexed with the node numbers. Let $X[1] = 0$, and let $X[x] = n$ for $x = 2 \ldots n$. Under these assumptions, what does the following algorithm do?

(Feel free to refer to similar or related algorithms in the curriculum.)

for $x \in V$
  for $(y, z) \in E$
    $X[z] = \min\{X[z], X[y] + 1\}$

Problem 6 You are to distribute courses among lecturers for the next semester. You have a list of courses to be held, and a set of lecturers to choose from. For each lecturer you have a list of courses he or she is competent to hold. The lecturers can also handle different workloads, and you are told how many courses each lecturer can bear to hold in a semester. The goal is to ensure that all courses have a lecturer without making any lecturer hold a course he or she is not competent to hold, and without making anyone hold more courses than their capacity.

a) (5 %) Describe how the problem can be solved as a flow problem.

You now discover that there are collisions between some of the courses (that is, they are to be held at the same time). You wish to avoid giving a lecturer two (or more) courses that collide. (If two courses collide, they are held at the same time and have the same duration. Therefore, you can assume that if course A collides with course B, and B collides with course C, then A will also collide with C.)

b) (5 %) Describe how you can handle the collisions and still solve the problem as a flow problem.