Department of Computer and Information Science

Examination paper for TDT4120 Algorithms and Data Structures

Academic contact during examination	Magnus Lie Hetland
Phone	918 51 949
Examination date	Dec 10, 2018
Examination time (from–to)	09:00–13:00
Support material code	D
Other information	The problem sheets are handed in, with answers in answer boxes under the problems
Language	English
Number of pages (front page excluded)	7
Number of pages enclosed	0

Informasjon om trykking av eksamensoppgave

Originalen er

1-sidig ☑ **2-sidig** ☐

sort/hvit ☑ **i farger** ☐

Skal ha flervalgskjema ☐

Checked by

Date Signature

Students will find the examination results in Studentweb. Please contact the department if you have questions about your results. The Examinations Office will not be able to answer these.

Questions for us?

We may only answer questions about possible errors, flaws or ambiguities in the problem texts.

We make one round and have limited time. Read all problems first and have any questions ready!

Please read this thoroughly

(*i*) Read the entire exam thoroughly before you start!

(*ii*) Write your answers in the answer boxes and hand in the problem sheet. Feel free to use a pencil! Or draft your answers on a separate piece of paper, to avoid strikouts, and to get a copy for yourself.

(*iii*) You are permitted to hand in extra sheets, if need be, but the intention is that the answers should fit in the boxes on the problem sheets. Long answers do not count positively.

Problems

5% 1. In a connected graph with $n \geq 1$ vertices and m edges, what is the least and greatest number of edges a spanning tree may have?

5% 2. In a Huffman code for an alphabet of $n \geq 1$ characters, what is the least and greatest length a codeword for a single character may have?

5% 3. Assume that Π is a predecessor matrix for the all-pairs shortest path problem. What does π_{ij} represent, if you assume $\pi_{ij} \neq \text{NIL}$?

5% 4. What is a cut of a flow network $G = (V, E)$?

5% 5. What is a vertex cover for a graph $G = (V, E)$?

5% 6. TABLE-INSERT has a worst-case running time of $\Theta(n)$, but an *amortized* running time of $O(1)$. What does that mean?

5% 7. In the n-rooks problem one has an $n \times n$ chessboard and is to place n rooks on the board, so there is exactly one rook in each row and one in each column. (A more thorough explanation may be found on page 7.) Your friend Klokland has found a solution to this problem, for a specific n—that is, a specific placement of the n rooks. Your friend Smartnes wants to know which of the possible solutions Klokland has found, but he won't tell her. Instead, she gets to guess, asking yes/no questions. How many questions will she need, in the worst case, if you can't assume anything about how smart Smartnes is? Give your answer in asymptotic notation. Explain your answer.

5% 8. In the 0-1-knapsack problem, assume that you have n objects and a knapsack capacity of m. What is the running time of the solution in the curriculum?

5% 9. If you insert the numbers $1, \dots, n^3$ into a binary search tree in random order, where $n = 2^k - 1$ for an integer $k \geq 1$, what is the expected height of the tree, as a function of k? Use Θ notation.

Q	1	2	3	4	5	6
	2	3	5	7	8	9

Figure 1: Example queue for problem 11. The values are inserted from the start, so Q.*head* = Q.*tail* = 1

Q	1	2	3	4	5	6
	7	8	9	2	3	5

Figure 2: Example queue for problem 11. The values are *not* inserted from the start. Q.*head* = Q.*tail* = 4

5% 10. Your friend Smartnes executes DFS on a DAG. Which kinds of edges could she get?

(Here *kinds* refers to the DFS edge classification.)

5% 11. Your friend Gløgsund has inserted n distinct numbers in increasing order into a FIFO queue Q, implemented as an array. The queue is full, so the array contains only the elements she has inserted. In other words, the elements either make up a single increasing segment, $\langle x_1, \ldots, x_n \rangle$ (see fig. 1), or two increasing segments, $\langle x_k, \ldots, x_n, x_1, \ldots x_{k-1} \rangle$ (see fig. 2), where $x_i < x_{i+1}$ for $i = 1 \ldots n-1$. Unfortunately, Gløgsund has forgotten where the queue begins and ends (i.e., Q.*head* og Q.*tail*). Describe an algorithm that determines this as efficiently as possible. What is the running time?

(A high-level description suffices, without concern for practical details or special cases.)

5% 12. In each iteration of the outermost loop of INSERTION-SORT, an operation is performed on A$[1\,..\,j-1]$ and A$[j]$. What is this operation, and what is the running time, as a function of j? Use O notation.

W	1	2	3	4
1	0	1	5	6
2	∞	0	−3	4
3	∞	∞	0	2
4	∞	∞	∞	0

Figure 3: Weight matrix for G, used in problem 14

$T^{(1)}$	1	2	3	4
1	1	0	1	0
2	1	1	1	0
3	0	0	1	0
4	0	1	0	1

Figure 4: Previous state, used in problem 15

5% 13. Your friend Lurvik has developed a new priority queue of which he is quite proud. It may be constructed in linear time, just like a binary heap, while Extract-Min and Decrease-Key have respective running times of $O(n^2)$ and $O(n^3)$, where n is the number of elements in the queue. What is the running time of Dijkstra if one uses Lurviks priority queue? Explain briefly.

(Note that Lurvik does *not* employ a series of calls to Insert to build his queue, but constructs it in linear time at the beginning of Dijkstra.)

5% 14. Let G be a weighted graph, defined by the weight matrix in fig. 3. Execute DAG-Shortest-Path on the graph, with 1 as the start node.

Fill in the distances to each node after each iteration in each row of the table below.

	1	2	3	4
0	0	∞	∞	∞
1				
2				
3				
4				

5% 15. You have executed one iteration of Transitive-Closure, and ended up with the matrix $T^{(1)}$ as shown in fig. 4. Execute the next iteration, and fill in the resulting values in the table below.

$T^{(2)}$	1	2	3	4
1				
2				
3				
4				

5% 16. What is the running time of the algorithm ALPHA, below, where $n \geq 1$ is an integer? Give your answer in Θ notation, as a function of n.

```
ALPHA(n)
for i = 1 to n
    for j = i to n
        for k = 1 to n
            print "Lurvik rulz!"
```

5% 17. What is the running time of the algorithm BETA, below, where $n \geq 1$ is an integer? Give your answer in Θ notation, as a function of n.

```
BETA(n)
if n ≥ 2
    m = ⌊n/2⌋
    BETA(m)
    BETA(n − m)
    for i = 1 to n^2
        print "Smartnes rocks!"
```

5% 18. Which problem does the algorithm DELTA, below, solve, where $n, m \geq 0$ are integers?

```
GAMMA(n, m)
if m == 0
    return n
else return GAMMA(n, m − 1) + 1
```

```
DELTA(n, m)
if m == 0
    return 0
else return GAMMA(n, DELTA(n, m − 1))
```

5% 19. Your friend Kvikstad is naming a set of computers. He wants the names to start with different letters, one beginning with A, one with B, etc. He has not yet decided which machine gets a name beginning with which letter, but he has some naming suggestions for each machine, and wants to select each name from those. Draw a flow network (on the next page) that finds a valid selection for him, based on the following table:

Machine	Possible names
1	Abhoth
2	Aletheia, Byatis
3	Aletheia, Byakhee, C'thalpa
4	Azathoth, Byakhee, Cthulhu, Dagon

Answer to problem 19:

5% 20. A knight is placed in square a1 on an $n \times n$ chessboard and is to be moved k times, or until such a move would have placed it outside the board. For each step, one of the 8 possible moves is to be chosen at random. Describe an algorithm based on dynamic programming that determines the probability that the knight is still on the board after k moves.

(A more thorough explanation may be found on the next page.)

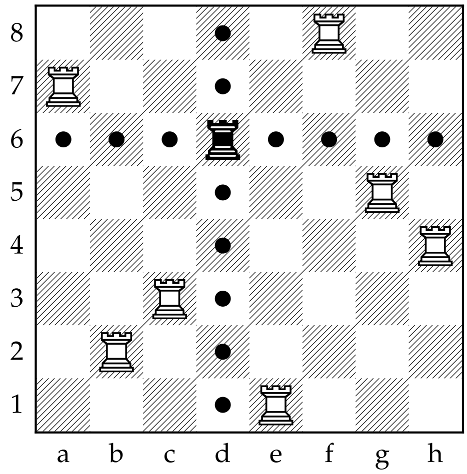

Figure 5: The n-rooks problem. The black rook threatens the squares indicated by black dots

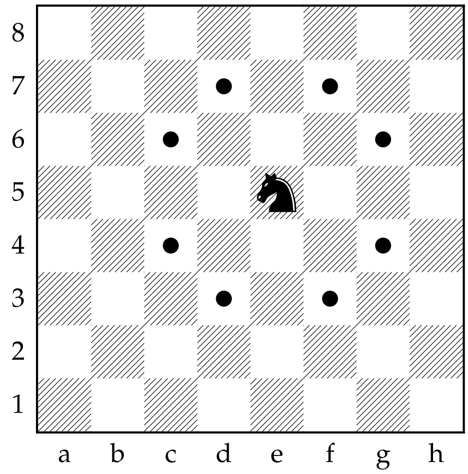

Figure 6: A knight may jump to one of eight squares, as indicated by black dots

About the chess problems

In the problems we consider, we permit arbitrarily large quadratic grids as chessboards, so an $n \times n$ chessboard consists of n rows and n columns.

In the n-rooks problem (see problem 7) we have an $n \times n$ chessboard and n rooks that are to be placed on the board, so that none of them *threatens* any other. A rook *threatens* all others in the same row or column, independently of color. (See fig. 5 for an example.) In other words, the problem consists of placing exactly one rook in each row and exactly one rook in each column. There are, of course, several correct solutions to this problem.

In the knight problem (see problem 20), we have only a single piece on an $n \times n$ chessboard. This piece is a *knight*, which may be moved to one of (at most) eight different positions, as shown in fig. 6. That is, it may be moved two squares up, down, to the right or to the left, and then one square to the side. For example, two squares to the right and then one square down, or two squares down and one to the right.

If the knight is near the edge of the board, some of these eight moves will no longer be permissible, as they would take the knight off the board. In our problem we still select among all eight (with uniform probability), and wish to see whether the knight does end up outside the board. If, for example, it is placed on square a1 (bottom left) and moves two squares up and one to the left, which is one of the eight possible moves, it has ended up outside, and we can no longer keep moving it.

We select each move independently of the others. Remember that the probability for two independent events A and B with respective probabilities $P(A)$ and $P(B)$ to both occur is the product of their probabilities, $P(A) \cdot P(B)$