

Midterm

TDT4120 Algoritmer og Datastrukturer

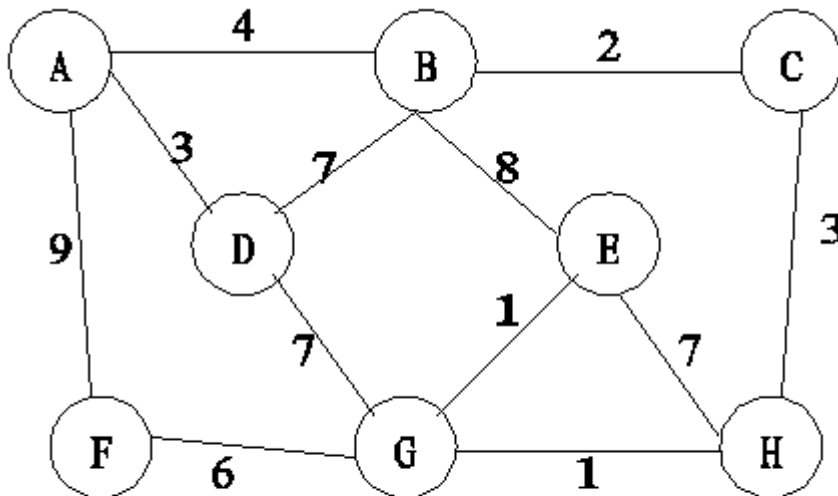
Wednesday, October 1st 2003

PS: This does not count on your final grade.

1. Algiator

- a) What does Big-Theta notation mean, as in $\Theta(n)$? (5 %)
- b) What is the worst case running time of quicksort? (5 %)
- c) What does it mean that an algorithm is *stable*? (5 %)
- d) Does a binary tree with height n always have 2^n leaf nodes? Why/why not? (5 %)

2. Spanning Tree



- a) Use a minimum spanning tree algorithm and find the minimum spanning tree in the graph. (10 %)
- b) Suppose that we multiply all weights in the graph with $\log(n)$, where n is the number of nodes of the graph. Which edges will then form the minimum spanning tree? (5 %)

3. Towers of Hanoi

Recall the famous game of Towers of Hanoi, many of you have good experience with it already. The picture shows an example how the game looks in reality:

The goal of the game is to move all rings from one column to another, which sound like an easy job, but the following rules must be obeyed:



- Only one ring can be moved in each step
- If ring A is larger than ring B, then ring A can not be placed above ring B.

It seems that the task can be solved with a simple recursive method that will generate instructions for all required single ring moves. That's what we are going to do now.

- a) Given is a piece of code in java that solves the problem of Towers of Hanoi. Your (10 %) task is to fill the lines A and B so that they work correctly. number represents the number of rings to be moved, etc.

```
public static void move(int number, int from, int to, int via){
    if(number>1){
        ???A????
    }
    System.out.println("Move ring from "+from+" to "+to);
    if(number>1){
        ???B????
    }
}
```

- b) Write a recurrence for the method you wrote in a). Let n denote the number of rings to move. You can assume that the operation of moving one ring from one column to another can be done in constant time. (5 %)
- c) Solve the recurrence in b) to find out in Θ -notation how many times do you need to move a single ring to solve the problem of Towers of Hanoi.

4. The Unlucky Number

The trainer Roger Mykodde is pretty strict at his trainings, but the true secret behind the victories in the elite league is that none of his football players wears his unlucky number 16 on the dress. But even more, it is also important that when the team stands in a row, one player next to another one, he cannot see a sequence of consecutive players whose dress numbers would sum up to his unlucky number. The team managers now bought some new players, and would like to make sure that there is no sequence of the player numbers that sums up to the unlucky number (the players of the team stand always in the same order before the training).

A cousin of the trainer who is colonel has a similar problem. He must avoid seeing the number 345678, either alone or as a sum of several numbers that follow one another. He is also very interested to get the Roger's algorithm, but since he works with many numbers, the time complexity should be minimal.

a) Find a solution for the problem described above. Given is a sequence of numbers (15 %) and the unlucky number. Find out whether the sequence contains a continuous subsequence that sums to the unlucky number. Sketch your idea behind the algorithm.

b) Write a pseudo code for your idea, you may use Java code if you want to. (10 %)

The method definition should be:

for pseudo code:

```
hasUnluckyNumber(A, unluckyNumber)
```

for Java:

```
public static boolean hasUnluckyNumber(int[] sequence, int unluckyNumber)
```

c) Find the asymptotic time complexity for your program in Θ -notation. (2,5 %)

d) Is your algorithm optimal (yes/no)? (2,5 %)

5. The Master Theorem

Solve the following recurrences using the master theorem:

a) $T(n) = 16T(n/2) + n^3 * \log(n)$ (5 %)

b) $T(n) = 25 * T(n/5) + n^2$ (5 %)

6. Bonus Question

Do you want to become a real Algiator? Then you should use this opportunity to solve our bonus assignment.

You have n points in the x - y -plane, $P_1 : (x_1, y_1), P_2 : (x_2, y_2), \dots, P_n(x_n, y_n)$. You will attempt to find a shortest path from P_1 to P_n , through a selection of the remaining $(n-2)$ points. The longest distance between two neighbouring points in this path should be as short as possible. (Imagine jumping from stone to stone, crossing a river, where the longest jump you make should be as short as possible).

a) Sketch the most efficient algorithm that will solve this shortest-path-problem by using Dijkstras algorithm (unmodified) as a subroutine. (0 %)

b) Find a more efficient algorithm to solve our shortest-path-problem by modifying the code in Dijkstra's algorithm. Show these modifications in detail. (0 %)