Algorithms Exam [1]

Oct. 19, 2010 **kl 14 - 18** **byggnad M**

Ansvarig:

Devdatt Dubhashi Tel. 073 932 2226 Rum 6479 EDIT

Points :	60	
Grades:	Chalmers	5:48, 4:36, 3:28
	GU	VG:48, G:24
	PhD students	G:36
Helping material :	course textbook, notes	

- Recommended: First look through all questions and make sure that you understand them properly. In case of doubt, do not hesitate to ask.
- **Answer all questions in the given space on the question paper (the stapled sheets of paper you are looking at). The question paper will be collected from you after the exam. Only the solutions written in the space provided on the question paper will count for your points.**
- Use extra sheets only for your own rough work and then write the final answers on the question paper.
- Try to give the most effciient solution you can for each problem - your solution will be graded on the basis of its corectness *and* efficiency. In particular a brute force solution will not get any credit.
- Answer concisely and to the point. (English if you can and Swedish if you must!) **Your solution will be graded both for correctness and efficiency - a faster algorithm will get more credit than a slower one.**
- Code strictly forbidden! Motivated pseudocode or plain but clear English/Swedish description is fine.

Lycka till!

[1] 2010 LP 1, INN200 (GU) / TIN090 (CTH).

Problem 1 Median [10] The input consists of two arrays $A[1 \dots n]$ and $B[1 \dots n+1]$ containing positive integers, all distinct and both in sorted order. Give a fast algorithm to find the median of all the $2n+1$ numbers i.e. a value $A[i]$ or $B[j]$ such that exactly half the numbers are less than this value and half greater. For example, if $A = [3, 12, 14, 44]$ and $B = [5, 17, 28, 31, 40]$, then the median is $17 = B[2]$. For full credit, your algorithm must run in time $O(\log n)$.

Problem 2 Bottlenecks [10] Consider a network of computers represented by a graph $G = (V, E)$: the vertices are computers and an edge represents a communication link between the two endpoints. Each edge e has a number c_e associated with it which is the maximum rate of data transmission it can support. You need to send data form your computer to your friend's computer at the maximum possible rate. If P is a path in G between the vertex u representing your computer and the vertex v representing your friend's, then the maximum rate of sending data along P is determined by the minimum rate c_e of an edge on the path P: this is called the *bottleneck* rate of the path P. Thus you want to find a path P between u and v with maximum bottleneck rate.

(a) [4 pts] Give a greedy algorithm to solve the problem.

(b) [3 pts] Argue that your algorithm is correct and optimal.

(c) [3 pts] What is the running time of your algorithm? Justify with the appropriate data structures.

Problem 3 Bitonics [10] Given an array $A[1 \ldots n]$ containing positive integers, the subarray $A[i \dots j]$ is *bitonic* if there is a k with $i \leq k \leq j$ such that

$$A[i] \leq A[i+1] \leq \cdots \leq A[k] \geq A[k+1] \geq \cdots \geq A[j-1] \geq A[j]$$

For example, if $A = [10, 2, 4, 5, 7, 6, 3, 8]$, then the subarrays $A[1 \dots 2]$ or $A[7 \dots 8]$, $A[4 \dots 6]$ or $A[2 \dots 7]$ are all bitonic. In this problem you have to develop a Divide-and-Conquer algorithm to find the maximum length of a bitonic subarray in a given input array A. In the example, it is 6.

(a) [2 pts] Describe the "Divide" step, and show what the recursive calls are.

(b) [5 pts] Describe the "Conquer" or "Combine" step: how to combine the solutions to the subproblems described in (a) to obtain a solution to the full problem.

(c) [2 pts] Write a recurrence for the running time for the algorithm.

(d) [1 pt] What is the solution to the recurrence in (c)?

Problem 4 Planning Cycle Trip [10] You are planning a cycle trip in the summer along the east coast starting at Stockholm and ending in Kiruna. You begin on day 1 at Stockholm and on day k you arrive at Kiruna, in between there are n towns numbered $1..n$ where you can rest for the night. Let us number Stockholm as 0, Kiruna as $n+1$. There is an array $d[1 \ldots n+1]$ such that the distance from Stockholm to city i is given by $d[i]$ and so the distance between succesive towns i and $i+1$ is given $d[i+1]-d[i]$. To rest for the night at the only "vandrahem" (youth hostel) in city i costs $c[i]$, for $i = 1 \cdots n$. Assume $c[0] = 0 = c[n+1]$. The total distance to cycle is $d[n+1]$ and you would like to spread this as evenly as possible between the k days, so that on average you cycle $\bar{d} := d[n+1]/k$ each day. However, this may not always be possible and sometimes you cycle a bit more and sometimes a bit less. If you cycle y km on a certain day, you penalty for this day is $(y-\bar{d})^2$. Thus if you cycle y_i km on day i and stay the night at vandrahem in city v_i, then your total cost is

$$\sum_{1 \leq i \leq k} \{c[v_i] + (y_i - \bar{d})^2\}.$$

Your goal is to find a schedule to cycle that minimizes this total cost.
In this problem, you develop a dynamic programing solution to the problem. Let $OPT(i,j)$ be the minimum cost for ending at town i on day j.

(a) [1 pt] In this notation, what is the final solution we want?

(b) [1 pt] What is the value of $OPT(0,j)$? $OPT(i,0)$?

(c) [3 pts] Write a recurrence for $OPT(i,j)$. (HINT: consider what is done on day j.)

(d) [2 pts] Using (b) and (c), implement the recurrence efficiently in pseudocode.

(e) [1 pt] What is the time and space complexity of your algorithm?

(f) [2 pts] How do you recover the actual schedule i.e where you show stop each night?

Problem 5 Closest Pair [10] The *element distintness* problem is to decide if a given set of input numbers $x_1, x_2, \cdots, x_n$ has all distinct elements i.e. $x_i \neq x_j$ for all $i \neq j$.

(a) [5 pts] Give a $O(n \log n)$ time algorithm to solve the problem.

It can be shown that *any* algorithm to solve this problem needs time $\Omega(n \log n)$.

(b) [5 pts] Assuming this, show that the algorithm we discussed in class (section 5.4 in the textbook) to find the closest pair of points in 2D is optimal i.e. *any* algorithm to find the distance between the closest pair of points in 2D needs time $\Omega(n \log n)$.

Problem 6 Profs for Dinner [10] You want to invite some Profs from the CSE Department for dinner (wonder why?!). As is well known, Profs are jealous of each other and end up not even on speaking terms. A grad student in the local sociology department created a database which is a two dimensional $n \times n$ matrix H (where n is the total number of CSE Profs) with $H[i, j] = 1$ if Prof i is on speaking terms with Prof j and 0 otherwise (there are many 0's but at least let's assume $H[i, i] = 1$ for all i!) To make a good party, you want to invite at least k Profs, but so that all are willing to talk to each other.

(a) [5 pts] Show that the problem of deciding if there is such a group of Profs is in $\mathcal{NP}$.

(b) [5 pts] Show that the problem is $\mathcal{NP}$-complete by giving a reduction from a problem (which is know to be $\mathcal{NP}$-complete).