Institut for Matematik og Datalogi
Syddansk Universitet

January 30, 2009
JFB

DM508 – Algorithms and Complexity – 2009
Lecture 1

Textbook and notes

Introduction to Algorithms, 2nd edition, by T. Cormen, C. Leiserson, R. Rivest, and C. Stein, MIT Press, 2001.

Extra notes (available in the bookstore): *DM508 Algoritmer og kompleksitet, Noter 3. kvartal 2009*. From the following sources:

- *Computer Algorithms: Introduction to Design and Analysis*, second edition, by S. Baase, Addison-Wesley, 1987.
- *Combinatorial Optimization: Algorithms and Complexity*, by C.H. Papadimitriou and K. Steiglitz, Prentice-Hall, 1982.

Format

Lectures will be in English. Please read the appropriate sections in the textbook or notes before coming to class and bring your textbook with you. There will both be assignments which you are required to turn in and other assignments which you should be prepared to discuss in the discussion sections (øvelserne), usually shortly after the relevant lecture. The discussion sections will be on Thursdays, 12:15–14 (weeks 6–12) in U148, and Tuesdays, 8:15–10 (weeks 7 and 9), in U9. The "instruktor" for the course is Nikolaj Blytsø. We expect that the discussion section will be in Danish, unless there are some foreign exchange students in the class.

The required assignments will be graded on a Pass/Fail basis, and satisfactory completion of all 3 assignments is required for a Pass. The assignments must all be turned in on time. (Note that if you send them to me by e-mail, you are responsible for them actually getting to me on time. Sending from IMADA's computers will help ensure that your e-mail is not delayed at an intermediate computer for many hours.) If you would like a receipt showing that you turned in your assignment, you may get one by submitting your assignment through Blackboard, in addition to turning it in as usual. (The Blackboard submission is only used for receipts.) To do this, login to Blackboard and find DM508. Choose "Course Tools" from the "Tools" menu on the left. Choose "Assignment hand in", fill out the form and end with "Submit". You can print out the receipt and you will get an e-mail. You must also turn in the assignment as usual. You may work in groups of 2 to 3 students if you wish. These 3 assignments must be approved in order for you to take the oral exam, so cheating on these assignments is viewed as cheating on an exam. You are allowed to

talk about course material with your fellow students, but working together on assignments with students not in your group is cheating. Using solutions you find elsewhere, such as on the Internet, is also cheating. You may do the assignments in either English or Danish, but if you write them by hand, please do so very neatly. You will be allowed to redo one of the assignments if it is not approved the first time (if one of your assignments is late, then you will have used up your only chance to redo any assignment).

The weekly notes and other information about the course are available through the World-wide Web. Use Blackboard or the URL:

http://www.imada.sdu.dk/~joan/dm508/

I have office hours 9:00–9:45 on Mondays and Thursdays.

There will be an oral exam at the beginning of April, 2009. Previous exam questions are available on DM508's homepage; an updated set of exam questions will be available later in the course (probably the same, except with randomized algorithms removed). You may do your exam in Danish if you wish (in most cases it is advisable to do it in Danish).

Lecture, February 2

We will begin with an introduction to the course. Lower bounds from section 2.4 of the first part of the notes will be discussed (part of this is also in section 8.1 of the textbook). We will also begin on sections 3.1, 3.2 and 3.3 of those notes.

Lecture, February 4

We will finish section 3.3 and will cover section 3.5 of the DM508 notes, plus median finding from chapter 9 (sections 9.2 and 9.3) in the textbook. We may also begin on NP-completeness, from chapter 34 in the textbook and the section by Papadimitriou and Steiglitz from the course notes.

Problems to be discussed on February 5

Do problems:

1. Do problems 3.2 (use Stirling's approximation - formula 3.17 from the textbook, and compare your result to the upper bound for merging, rather than to the lower bound mentioned) and 3.10 from pages 140 and 141 of the notes.

2. Prove a lower bound for merging two lists of lengths n and m which meets the upper bound of $n + m - 1$.

3. From the following pages of that book by Baase:

Consider the problem of determining if a bit string of length n contains two consecutive zeros. The basic operation is to examine a position in the string to see if it is a 0 or a 1. For each $n = 2, 3, 4, 5$ either give an adversary strategy to force any algorithm to examine every bit, or give and algorithm that solves the problem by examining fewer than n bits.

and

a. You are given n keys and an integer k such that $1 \leq k \leq n$. Give an efficient algorithm to find *any one* of the k smallest keys. (For example, if $k = 3$, the algorithm may provide the first-, second- or third-smallest key. It need not know the exact rank of the key it outputs.) How many key comparisons does your algorithm do? (Hint: Don't look for something complicated. One insight gives a short, simple algorithm.)
b. Give a lower bound, as a function of n and k, on the number of comparisons needed to solve this problems.

4. From Baase's textbook: Suppose $L1$ and $L2$ are arrays, each with n keys sorted in ascending order.

   a. Devise an $O((\lg n)^2)$ algorithm (or better) to find the nth smallest of the $2n$ keys. (For simplicity, you may assume the keys are distinct.)

   b. Give a lower bound for this problem.

5. Design and analyze an efficient algorithm to find the third largest item in an array.

6. Consider a company with a customer service department consisting of an automatic system and two representatives. The company has customers in three cities, Detroit, Chicago, and San Francisco. A request for service requires one of the service representatives to be in that city to service it. Each city has an apartment for the service representative, who will stay there until called to another city. Assume the flight between Chicago and Detroit costs only a fraction $1/d$ of what the flight between Chicago and San Francisco costs, and that it is impossible to fly from Detroit to San Francisco without changing planes in Chicago and paying the sum of the ticket costs from Detroit to Chicago and from Chicago to San Francisco. (Assume that all costs are symmetric, so that it costs exactly as much fly the opposite direction between two cities.) The company wishes to minimize the amount it spends on plane tickets. Assume in what follows that there is initially one service representative in Detroit and one in San Francisco. In the following, assume that if the next request for service is in a city where there is already a service representative, then no service representative flies anywhere, and this costs nothing.

   (a) Show that in the worst case, any algorithm has cost at least $n \cdot f$ on a sequence of requests of length n, where f is the cost of the cheaper flights.

(b) Consider the Greedy algorithm which always requires the closer representative (note that the closer pair of cities has the lower cost airfare) to travel (when travel is necessary). Also consider the algorithm Dummy which, on requests in Chicago always sends the representative which is in San Francisco (when travel is necessary), but otherwise sends the closer representative. Show that for any constant c, there exists a sequences where Dummy pays a factor more than c times what Greedy pays and another sequence where Greedy pays a factor more than c what Dummy pays.

Assignment due Wednesday, February 18, 10:15

Note that this is part of your exam project, so it must be approved in order for you to take the exam in March, and you may not work with or get help from others not in your group. You may work in groups of two or three. You may write your solutions in English or Danish, but write very neatly if you do it by hand.

1. Consider the problem of Sorting by Reversals. You are given a permutation of the numbers from 1 to n in an array, A. The operation you have is to choose two indices, i and j, and to reverse the elements in the subarray from $A[i]$ to $A[j]$, inclusive. The objective is to end with a sorted array. For example, given $A = [8, 6, 4, 2, 7, 5, 3, 1]$, the first operation could be $(3, 6)$, resulting in $A = [8, 6, 5, 7, 2, 4, 3, 1]$. Then doing the operations $(2, 4), (3, 4), (5, 7), (5, 6), (1, 8)$ would finish sorting the array.

   a. Give an algorithm which sorts the array in $O(n)$ operations.

   b. Prove that any algorithm needs at least $\Omega(n)$ operations in the worst case.

   c. Why doesn't the information-theoretic lower bound of $\Omega(n \log n)$ apply here?

2. Consider the following algorithm:

```
Input: An integer array A of length n
if A[1] ≤ A[2]
    then second ← A[1]; max ← A[2]
    else second ← A[2]; max ← A[1]
end if
for i = 3 to n do
    if A[i] > second
        then second ← A[i]
    end if
end for
if (max < second)
    then temp ← second; second ← max; max ← temp;
end if
return(max,second)
```

Give two proofs that this is not an algorithm for finding the largest and next largest elements of an array. One proof should refer to a lower bound result you already know. The other should not.