Department of Mathematics and Computer Science October 4, 2010
University of Southern Denmark JFB, KSL

DM205 – On-Line Algorithms – Lecture 10

Lecture, October 4

- Borodin & El-Yaniv, mostly overview of Chapter 6, but with proofs in Section 6.4.2 done carefully.

Lecture, October 7

- Borodin & El-Yaniv, overview of Chapters 7 and 8, skipping most proofs other than Theorem 8.7.

Lecture, October 12

- Borodin & El-Yaniv, Sections 10.0–10.2, 10.4, 10.6.

Exercises, October 14

All references are to the textbook by Borodin & El-Yaniv unless otherwise stated.

1. Give an upper bound on the complexity of the dynamic programming procedure used for computing the cost of an optimal offline algorithm for the k-server problem when the request sequence is of length n.

   For the special case of a uniform metric space (paging) a faster algorithm exists; one must compute the cost of LFD. What is the best upper bound you can give on that?

2. Define and analyze a lazy version of DC for paging.

3. Exercise 10.1. For ease of computation, define such a mesh to be one with distance $\sqrt{N}$ from corner to corner along a side, i.e., with $\sqrt{N}+1$ points along a side.

4. Consider the following on-line problem: We have one processor and jobs arrive over time. Job J_j with processing time p_j arrives at time r_j. A job can be assigned to run on the processor when it arrives or any time after that. It can also be started on the processor, terminated

at some point, and restarted (not resumed; it must be started from scratch) at some later point. No two jobs may be running at the same time. The goal is to minimize total completion time. Let C_j denote the completion time of job j. The total completion time is $\sum C_j$. Note that we start the clock at time 0 and the completion time of a job is simply the time read off the clock when the job completes. Thus, it is neither its running time nor its flow time (time from release to completion).

Use Yao's principle (the second component of the max-expression of Theorem 8.5 is useful here) to prove a lower bound on the competitive ratio of any randomized algorithm for this problem. Consider the following probability distribution on request sequences: At time 0, a job with processing time 1 arrives. At time $\frac{1}{3}$, two jobs with processing time 0 arrive. At time 1, with probability p, an additional collection of job arrives. Thus, with probability $1-p$ no further jobs arrive. The collection of jobs that arrive with probability p consists of 10 jobs with processing time 0 and four jobs with processing time 1.

5. You're following a course on online algorithms on a less regulated university than SDU. You don't know for how long the course will run; your professor will simply tell you when all students have learned enough. With regards to the textbook, you have two options: you can buy it for k\$, or you can borrow it every night from a fellow student, who wants 1\$ per day for the trouble. Decide if and when to buy the textbook to get the best possible competitive ratio (against OPT who knows in advance for how long the course will run).

   This is perhaps the simplest possible problem of searching for the optimal *time* to make a (financial) decision. This is important in all sorts of trading problems for stocks, bonds, commodities, etc.

6. You parked your car on the side of the road through a desert (with two wheels on the pavement). Then you headed out for a walk. You got lost and it got dark – pitch black! Luckily, you suddenly felt that you were back on the road again. The only problem now is that you have no idea if the car is to the left or the right. Devise and analyze a competitive algorithm for finding the car (against OPT who knows if the car is to the left or the right). You remember on which side of the road the car is parked, so you will bump into it when you get there, and, thus, not accidentally miss it. The car is at least one step away from your starting point.

This is perhaps the simplest possible problem of searching for the optimal *location*. This is important in rescue operations, oil drilling, etc.