Prof. Dr. Susanne Albers
Dr. Dimitrios Letsios
Dario Frascaria
Lehrstuhl für Theoretische Informatik
Fakultät für Informatik
Technische Universität München

Problem set 1
April 20, 2015
Summer Semester 2015

Online and Approximation Algorithms

Due April 27, 2015 before class!

Exercise 1 (Ski Rental - 10 points)

The ski rental problem is defined as follows: Assume that renting a pair of skis costs 1 per day while buying a pair of skis costs b. Every day we have to decide, in an online fashion whether we want to continue renting skis for another day or buy a pair of skis. At some unknown time D, we will break our leg and have to quit skiing. Our goal is to minimize the cost of skiing. Show that there is no deterministic online algorithm with a competitive ratio better than $2 - \frac{1}{b}$.

Exercise 2 (List Scheduling on Unrelated Machines - 10 points)

In class, we presented an online greedy algorithm for the problem of scheduling n jobs on m identical machines. In this exercise, we consider the same problem in a more complex environment with m *unrelated* machines $M_1, M_2, \ldots, M_m$. Now, the processing time of a job J_j, $1 \leq j \leq n$, depends on the machine by which it is executed and it is not necessarily the same for every machine. Specifically, if job J_j is executed by machine M_i, then its processing time is $p_{i,j}$. The greedy algorithm presented in class for identical machines can be easily extended for unrelated machines as follows: Schedule each job J_j on a machine M_i that results in the schedule with minimum makespan. Prove that the competitive ratio of Greedy is not smaller than $\dfrac{m}{1+\varepsilon}$, where ε is an arbitrary small constant.

(Hint: There exists an example where each job has finite processing time only on two machines and infinite processing time on all the other machines.)

Exercise 3 (First-in First-out - 10 points)

Recall that FIFO is the online paging algorithm that evicts the page that has been in fast memory for the longest time. Prove that FIFO is k-competitive, where k is the number of pages that fit in fast memory.

Exercise 4 (Demand Paging - 10 points)

Paging algorithms that do not evict pages unless there is a page fault are called *demand paging*. Prove that any paging algorithm can be modified to be demand paging without increasing the overall number of memory replacements on any request sequence.