15-750 — Graduate Algorithms — Spring 2006

Miller and Derryberry

Assignment 6 Due date: Friday, May 5, 2006.

Some Reminders:

- Read the Policies section on the course website before you start working on this assignment.
- You may work in **groups of size up to 3** for this problem set if you wish. However, **you should write up your solutions separately**. That is, collaboration should be limited to talking about the problems, so that your writeup is written entirely by you and not copied from your partner. In addition, state whether you worked alone and **list all collaborators**.
- Please refrain from consulting external materials when solving these problems. This does not apply to looking up standard inequalities, definitions, and such things (e.g., $(1 + 1/x)^x \leq e$ or Stirling's Formula).
- Please **submit both an electronic version, as well as a hard copy of your solutions** at the beginning of class on the due date.
- In all problems, it is implicit that you should show that your answer is correct, even when this is not explicitly stated.
- If you have questions, contact the course staff.

1 Maximum 2CNF

Let 2CNF denote the set of all boolean formulas in conjunctive normal form with at most two literals per clause. Let 2CNFSat be the subset of 2CNF formulas that are satisfiable. Define the set Max-2CNF as follows

$$\text{Max-2CNF} = \{(\phi, k) \mid \phi \in \text{2CNF, and } \phi \text{ has a truth assignment making at least } k \text{ clauses true}\}$$

(a) (10 points) Show that the language 2CNFSat is polynomial time decidable.

(b) (15 points) Show that deciding membership in Max-2CNF is NP-Complete.

2 Randomized Move-To-Front

In class we considered an on-line algorithm problem called The List Update Problem.

In The List Update Problem we have a list containing a fixed set of n elements and a sequence of accesses to elements of the list. As in the notes, the cost of the operation Access(x) is the index of

x in the list. In addition, at any time, an algorithm is permitted to swap 2 adjacent members of the list at a cost of one per swap.

Consider the following on-line algorithm for The List Update Problem:

Random Move To Front (RMTF): After an access to an element x, flip a fair coin. If the outcome is heads, do a series of swaps to move x to the front of the list, otherwise do no swaps.

(a) (15 points) Show that the expected competitive ratio for RMTF is 3 for the list-update problem.

Hint: Make sure you make use of random variable and linearity of expectation.

3 Parallel Machine Scheduling

In the **parallel-machine-scheduling problem** we are given n jobs $J_1, \ldots, J_n$ with nonnegative processing times $p_1, \ldots, p_n$ for jobs, each to be scheduled on one of m identical machines. Once a job is started it must run until completion and no other job may be run on that machine until the first one is completed. Thus, a **schedule** is a map from jobs to pairs consisting of a start time and a machine such that no 2 jobs will overlap on the same machine. Let C_k be the completion time for the kth job. We define $C_{max} = \max_{1 \le j \le n} C_j$ to be the **makespan** of the schedule. The goal of the parallel-machine-scheduling problem is to schedule the jobs so as to minimize the makespan.

Consider three scheduling algorithms:

Greedy: Schedule the jobs in the order they are given and on the first available machine.

Shortest-First: Sort the jobs in non-decreasing order by their processing times. Run Greedy on this newly sorted list.

Longest-First: Sort the jobs in non-increasing order by their processing times. Run Greedy on this newly sorted list.

The goal of this problem is to determine how close these algorithms' makespans are to the optimal makespan. Let C_{opt} be the optimal makespan for an instance of the parallel-machine-scheduling problem. In order to determine the approximation quality of the schedules we need to find lower bounds on C_{opt}.

(a) (2 points) Briefly show that $p_{max} = \max_{1 \le k \le n} p_k$, the maximum-sized job, is a lower bound on C_{max}.

(b) (2 points) Briefly show that $avg_m = \frac{1}{m} \sum_{1 \le k \le n} p_k$, the average machine load, is a lower bound on C_{max}.

(c) (6 points) Show that Greedy and Shortest-First are 2-approximation algorithms for minimizing the makespan.

(d) (10 points) Show that for any $\epsilon > 0$, Shortest-First is at best a $2 - \epsilon$-approximation algorithm.

(e) (10 points) Show that Longest-First is a 3/2-approximation algorithm.

Hints: 1) Let J_t be a job with largest completion time in the Longest-First schedule. Show that if $2p_t \le p_1$ the $C_{max} \le 3/2 C_{opt}$.

2) Consider those problems where job J_t is scheduled by Longest-First at time zero.