Chair of Computer Science I
Algorithms and Complexity
University of Würzburg

Würzburg, December 5, 2024

Prof. Dr. Alexander Wolff
Klaus Biehler

Homework Assignment #7
Approximation Algorithms (Winter Semester 2023/24)

We consider two scheduling problems on multiple machines.

Exercise 1 – Identical machines

First we look at scheduling on identical machines. Here we are given n jobs to be processed, and there are m identical machines (running in parallel) to which jobs may be assigned. Each job $j = 1, \dots, n$ must be processed on one of these machines for p_j time units without interruption, and each job is always available. Each machine can process at most one job at a time. The aim is to complete all jobs as soon as possible; that is, if job j completes at time C_j (the schedule starting at time 0), then we wish to minimize $C_{max} = \max_{j=1,\dots,n} C_j$, which is called the *makespan* or *length* of the schedule.

Consider the following algorithm, called *list scheduling*. Look at the list of all jobs (in some arbitrary order) and greedily assign each job to the machine that, in the partial schedule as constructed so far, finishes soonest.

a) Show that list scheduling has approximation factor 2. **[3 points]**

Now we sort the list of jobs in order of non-increasing processing time: this is called the *longest processing time rule*.

b) Show that this version of list scheduling is optimal if each job has length strictly larger than one-third the optimal makespan C^*_{max}. **[3 points]**

c) Show that this algorithm has approximation factor $4/3$. **[4 points]**

Exercise 2 – Related machines

Next we consider another scheduling variant. Now each machine i has an associated integer speed s_i, and it takes p_j/s_i units of time to process job j on machine i. Assume that the machines are numbered from 1 to m and ordered such that $s_1 \geq s_2 \geq \cdots \geq s_m$. These are called *related* machines.

A ρ-*relaxed decision procedure* for a scheduling problem is an algorithm such that given an instance of the scheduling problem and a deadline D, the algorithm either produces a schedule of length at most $\rho \cdot D$ or it correctly states that no schedule of length D is possible for the instance.

a) Show that given a polynomial-time ρ-relaxed decision procedure for the problem of scheduling related machines, one can produce a ρ-approximation algorithm for the problem.

   Hint: Use binary search with a lower and an upper bound. How close do they need to get in order to get the approximation guarantee? **[3 points]**

We will now look at a 2-relaxed decision procedure for scheduling related machines. Consider the following variant of the list scheduling algorithm. Given deadline D, we label every job j with the slowest machine i such that the job could complete on that machine in time D; that is, $p_j/s_i \leq D$.

b) What to do if such a machine does not exist? **[1 point]**

If machine i becomes idle at a time D or later, it stops processing. If machine i becomes idle at a time before D, it takes the next job of label i that has not been processed, and starts processing it. If no job of label i is available, it looks for jobs of label $i+1$; if no jobs of label $i+1$ are available, it looks for jobs of label $i+2$, and so on. If no such jobs are available, it stops processing. If not all jobs are processed by this procedure, then the algorithm states that no schedule of length D is possible.

c) Show that this algorithm is a polynomial-time 2-relaxed decision procedure. **[6 points]**

Please hand in your solutions via WueCampus until 10:00 on Friday, December 13.