Weekplan: Approximation Algorithms

Inge Li Gørtz

References and Reading

[1] The Design of Approximation Algorithms, Williamson and Shmoys, Cambridge Press, section 1.1, 2.1, 2.2, and 2.3.

[2] A unified approach to approximation algorithms for bottleneck problems, D. S. Hochbaum and D. B. Shmoys, Journal of the ACM, Volume 33 Issue 3, 1986.

We recommend reading [1] in detail before the lecture. [2] provides background on the k-center problem.

Exercises

1 [w] **Piazza** Enroll in the Piazza group for the course.

2 Longest processing rule Prove that for any input where the processing time of each job is more than a third of the optimal makespan, LPT computes an optimal schedule.

3 Tight examples for LPT and local search Give almost tight examples for the LPT and the local search algorithm for scheduling on parallel identical machines. That is, for LPT give an example showing that LPT can produce a schedule that is a factor $(4/3-1/3m)$ from optimum. Similarly, give an example where the schedule produced by local search is a factor $(2-1/m)$ from optimum.

4 Precedence constraints (exercise 2.3 in [1]) We consider scheduling jobs on identical machines as in Section 2.3, but jobs are now subject to precedence constraints. We say $i \prec j$ if in any feasible schedule, job i must be completely processed before job j begins processing. A natural variant on the list scheduling algorithm is one in which whenever a machine becomes idle, then any remaining job that is available is assigned to start processing on that machine. A job j is available if all jobs i such that $i \prec j$ have already been completely processed. Show that this list scheduling algorithm is a 2-approximation algorithm for the problem with precedence constraints.

5 The k-supplier problem The *k-supplier* problem is similar to the k-center problem, but the vertices are partitioned into *suppliers* $F \subseteq V$ and *customers* $C \subseteq V$. The goal is to find k suppliers such that the maximum distance from a customer to a supplier is minimized. Give a 3-approximation algorithm for the k-suppliers problem.

6 Metric k-clustering Give an 2-approximation algorithm for the following problem.

Let $G = (V, E)$ be a complete undirected graph with edge costs satisfying the triangle inequality, and let k be a positive integer. The problem is to partition V into sets $V_1, \ldots, V_k$ so as to minimize the costliest edge between two vertices in the same set, i.e., minimize

$$\max_{1 \le i \le k,\, u,v \in V_i} c(u, v).$$

7 Scheduling on related parallel machines (exercise 2.4 in [1]) In this problem, we consider a variant of the problem of scheduling on parallel machines so as to minimize the length of the schedule. Now each machine i has an associated speed s_i, and it takes p_j/s_i units of time to process job j on machine i. Assume that machines are numbered from 1 to m and ordered such that $s_1 \geq s_2 \geq \cdots \geq s_m$. We call these related machines.

7.1 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 either produces a schedule of length at most $\rho \cdot D$ or correctly states that no schedule of length D is possible for the instance. 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.

7.2 Consider the following variant of the list scheduling algorithm, now for related machines. Given a 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, $p_j/s_i \leq D$. If there is no such machine for a job j, it is clear that no schedule of length D is possible. 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.

Prove that this algorithm is a polynomial-time 2-relaxed decision procedure.