Chapter 4: Deterministic Rounding of Linear Programs

(cp. Williamson & Shmoys, Chapter 4)

Minimizing Sum of Completion Times on a Single Machine

Given: jobs with processing time $p_j > 0$, release date $r_j \geq 0$, $j = 1, \dots, n$.

Task: Schedule the jobs nonpreemptively on a single machine; minimize the *total completion time* $\sum_{j=1}^{n} C_j$.

Remarks:

- This problem is known to be strongly NP-hard.
- The preemptive relaxation, however, can be solved efficiently.

Shortest Remaining Processing Time (SRPT) Rule

- At any point in time, process an available and uncompleted job with shortest remaining processing time.

Theorem 4.1.

The SRPT Rule finds an optimal preemptive schedule in time $O(n \log n)$.

Proof: Use an exchange argument. □

Converting Preemptive into Nonpreemptive Schedule

Idea: Use optimal preemptive solution to get good nonpreemptive solution.

Algorithm

1. compute optimal preemptive schedule with job completion times C_j^P;
2. sort jobs such that $C_1^P < C_2^P < \cdots < C_n^P$;
3. schedule all jobs nonpreemptively and as early as possible in this order;

Step 3: set $C_1 := r_1 + p_1$; for $j = 2$ to n set $C_j := \max\{r_j, C_{j-1}\} + p_j$;

Lemma 4.2.

For each job $j = 1, \ldots, n$, it holds that $C_j \leq 2 \cdot C_j^P$.

Proof:... □

Theorem 4.3.

The algorithm above is a 2-approximation algorithm.

Proof:... □

Minimizing the Weighted Sum of Completion Times

Given: As before, but now all jobs j also have a weight $w_j \geq 0$.

Task: Minimize the total *weighted* completion time $\sum_{j=1}^n w_j C_j$.

Remarks:

- Unfortunately, already the weighted preemptive problem is NP-hard.
- Thus, instead of preemptive relaxation use LP relaxation.

$$\begin{aligned} \min \quad & \sum_{j=1}^{n} w_j C_j \\ \text{s.t.} \quad & C_j \geq r_j + p_j && \text{for all jobs } j = 1, \ldots, n, \\ & \sum_{j \in S} p_j C_j \geq \tfrac{1}{2} p(S)^2 && \text{for all } S \subseteq \{1, \ldots, n\}. \end{aligned}$$

Lemma 4.4.

The completion times C_j of a feasible schedule satisfy the LP constraints.

Proof:... □

Scheduling in Order of LP Completion Times

Lemma 4.5.

Despite the exponential number of constraints, an optimal solution C^* to the LP relaxation can be computed in polynomial time.

Proof:. . . □

Algorithm

1. compute optimal solution C^* to the LP relaxation;
2. sort jobs such that $C_1^* < C_2^* < \cdots < C_n^*$;
3. schedule all jobs nonpreemptively and as early as possible in this order;

Theorem 4.6.

The algorithm above is a 3-approximation algorithm.

Proof:. . . □