18/20

Advanced Algorithms UE 8

by Maarten Behn
Group 5

Exercise 8.1 (Single-Machine scheduling with Release Dates) (7 Points)

> Consider the preemptive single-machine scheduling problem $1|r_j, pmtn|\sum Cj$, in which a set of jobs,
> each with a processing time $p_j \geq 0$ and a release date $r_j \geq 0$ must be scheduled preemptively on a single machine with the objective to minimize the total completion time.
> Show that the Shortest Remaining Processing Time (SRPT) rule is optimal.
> SRPT schedules at any moment in time the job with the shortest remaining processing time.

Helper definition

Let $l_i(t)$ the processing time that is left at time t for Job i.

6/6

Proof

Take a optimal solution S^* that does not follow SRPT.
Let t be the earliest time where S^* deviates from SRPT.
At time t let:

- J_i the job that SRPT would process (shortest remaining processing time among released jobs)
- J_j the job that is being processed in S^*, where $J_j \neq J_i$
- $r_i, r_j \leq t$

- $l_i(t) \leq l_j(t)$

Construct a new schedule S' by modifying S^* as follows:

- Let $\delta \geq 0$ be a small time span.
- Pause J_j and process J_i in $[t, t+\delta]$ then continue with J_j.
  δ is small enough that no other job is released during $[t, t+\delta]$, and that neither J_i nor J_j finishes in that time.

Let C_i^* be the completion time of J_i in S^*
Let C_j^* be the completion time of J_j in S^*
then
$C_i' = C_i^* - \delta$
$C_j' = C_j^* + \delta$

$\Delta = (C_i' + C_j') - (C_i^* + C_j^*) = (-\delta + \delta) = 0$

Thus, the total completion time remains unchanged.
However, we have made the schedule more SRPT-like without making the objective worse.

By repeating this local exchange at every deviation from SRPT we can transform S^* into a schedule that exactly follows SRPT, without increasing $\sum C_j$ at any point.
Therefore, SRPT must also be optimal.

Exercise 8.2 (Makespan Scheduling via LP) (7 Points)

> We have seen in class that the scheduling problems $P|pmtn|C_{max}$ and $P|pmtn, r_j|C_{max}$ can be solved optimally in polynomial time. Recall that in these problems, we are given a set of jobs J, each with processing time $p_j \geq 0$ and possibly a release date $r_j \geq 0$, and m identical parallel machines to execute these jobs.

> The task is to find a feasible schedule, possibly using preemption, such that the makespan is minimized.
>
> Show how to solve the problem with release dates using linear programming (LP).
>
> Hint: Formulate the subproblem of assigning a feasible amount of jobs' processing to time intervals via an LP, given that you knew the optimal makespan.

LP for $P|pmtn, r_j|C_{max}$

Let $T \geq 0$ be the processing time of a minimal makespan.
5/7 Let T_p be a time span such that

- $\exists k \in \mathbb{N}\ T = T_p \cdot k$ (T can be split into k time spans T_p)
- $\forall j \in \{1, \ldots, n\}\ \exists l_j \in \mathbb{N}\ r_j = T_p \cdot l_j$ (And every release time sits at the start of one of these time spans)
  Let $t = 0, \ldots, k-1$ (The index of the time span of length T_p)

You have to compute T, you cannot set up the program by assuming you already know T.
Why do such numbers exist?

Variables

$x_{j,t} \in [0, 1]$ the amount a job j is processed in time span t

Goal

Find a valid distribution of $x_{j,t}$

Constraints

Every Job gets processed fully:

$$\sum_{t=l_j}^{n} x_{j,t} = p_j \;\; \forall j \in \{1, \ldots, n\}$$

At a t no more that m time spans of jobs get processed:

$$\sum_{j=1}^{n} x_{j,t} \leq m \;\; \forall t \in \{0, \ldots, k-1\}$$

A job is only processed after its release:

$$x_{j,t} = 0 \;\; \forall j \in \{1, \ldots, n\} \;\; \forall t \leq l_j$$

No negative:

$$x_{j,t} \geq 0 \;\; \forall j \in \{1, \ldots, n\} \;\; \forall t \in \{0, \ldots, k-1\}$$

You also have to make sure no job runs parallel to itself

Find minimal makespan

Perform a binary search for T in the interval of
$max_{\forall j \in \{1,\ldots,n\}}(r_j) \leq T \leq \sum_{j=1}^{n} \frac{p_j}{m}$
The smallest valid T is the optimal solution.

$P|pmtn|C_{max}$

There is reduction of $P|pmtn|C_{max}$ to $P|pmtn, r_j|C_{max}$ by setting all $r_j = 0$.
Therefore the optimal solution in $P|pmtn, r_j|C_{max}$ is also optimal for $P|pmtn|C_{max}$

Exercise 8.3 (Assignment Problem) (6 Points)

> In class we have reduced the unrelated-machine scheduling problem $R||C_{\max}$ to the problem of finding a perfect matching of minimum cost.

> Model the scheduling problem as a integer linear program.
> Argue that your formulation is correct.

Variables:

6/6 $x_{ij} \in \{0,1\}$ (is job j is assigned to machine i)
$C_{\max} \in \mathbb{R}$ (objective to minimize)

Optimization function

$min(C_{\max})$

Constraints

Each job is assigned to exactly one machine

$$\sum_{i=1}^{m} x_{ij} = 1 \;\; \forall j \in \{1, \dots, n\}$$

Load on each machine cannot exceed the $C_{\max}$:

$$\sum_{j=1}^{n} p_{ij} \cdot x_{ij} \leq C_{\max} \;\; \forall i \in \{1, \dots, m\}$$

Binary constraints:

$$x_{ij} \in \{0,1\} \;\; \forall i \in \{1, \dots, m\} \;\; \forall j \in \{1, \dots, n\}$$

Correctness Argument

Each job is assigned to exactly one machine (Constraint 1)
-> Feasibility
Each machine's total assigned job processing time does not exceed $C_{\max}$ (Constraint 2)

-> Machine load respected

The LP minimizes $C_{\max}$, which is goal in the scheduling problem.