Minimize Schedule Delay

CSCI 3100

Schedule to Minimize Delay

Input: a set of jobs, S, where each job “i" has a length t(i) and deadline d(i).

Output: schedule of all jobs on the same resource such that each job is delayed as little as possible

Example: schedule of doing your homework.

Delay

A job "i" is delayed if it is finished after the deadline: f(i) > d(i)

The **lateness** of a job is max(0, f(i) – d(i))

The lateness of a **schedule** is max_i = max(0, f(i) – d(i))

Problem statement

Input: {(t(i), d(i)), 1≤ i ≤ n)

Optput: {s(i), 1 ≤ i ≤ n) of start time such that:

- max_i (max (0, s(i) + t(i) – d(i))) is minimized

Template for Greedy Algorithm

Question: in what order should we schedule the jobs?
- Shortest length (increasing order of length t(i))?
- Shortest slack time (increasing order of d(i) – t(i))?
- Earliest deadline (increasing order of d(i))?

Earliest Deadline First

Order the jobs in order of their deadlines
Assume for simplicity of notation that $d_1 \leq \ldots \leq d_n$
Initially, $f = s$
Consider the jobs $i = 1, \ldots, n$ in this order
    Assign job i to the time interval from $s(i) = f$ to $f(i) = f + t_i$
    Let $f = f + t_i$
End
Return the set of scheduled intervals $[s(i), f(i)]$ for $i = 1, \ldots, n$

- Proof of correctness: use exchange argument
- Gradually modify optimal schedule O until it is the same as the schedule A produced by this algorithm

Properties of Schedules

A schedule has an *inversion* if a job i with deadline $d(i)$ is scheduled before a job j with an earlier deadline $d(j)$, i.e., $d(j) < d(i)$ and $s(i) < s(j)$.

Claim 1: The algorithm produces a schedule with no inversions and no idle time.

Claim 2: All schedules with no inversions and no idle time have the same lateness.

Claim 3: There is an optimal schedule with no idle time

Claim 4: There is an optimal schedule with no inversions and no idle time.

Claim 5: The greedy algorithm produces an optimal schedule (follows from 1, 2, and 4)

Proving claim 4

Claim 4: There is an optimal schedule with no inversions and no idle time.

Approach: Start with an optimal schedule O and use an *exchange argument* to convert O into a schedule that satisfies Claim 4 and has lateness not larger than O.