Oct 12

Scheduling to minimize max lateness

Input: n intervals, interval $i \in [n]$

d_i — deadline; t_i — duration

Output: $\forall\, i \in [n]$ compute $s(i)$

$\Rightarrow$ i is scheduled $[s(i), s(i)+t_i)$ = $[s(i), f(i))$

(no overlaps)

Goal: minimize max lateness

l_i (lateness of job i) $= \max(0, f(i) - d_i - 1)$

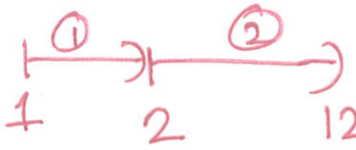
For a schedule S, max lateness $L(S) = \max_i l_i$

Example 1: $n = 2$

$t_1 = 1$, $d_1 = 100$

$t_2 = 10$, $d_2 = 10$

Schedule 1: (schedule by duration)

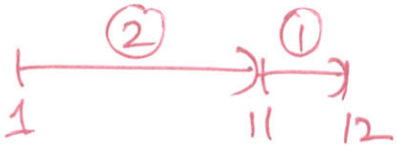


$l_1 = \max(0, 1-100) = 0$

$l_2 = \max(0, 12-10-1) = 1$.

$L = 1$

Schedule 2:



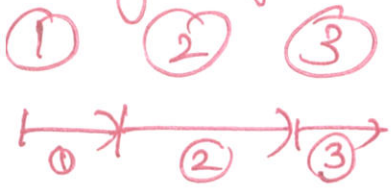
$l_1 = \max(0, 11-100) = 0$

$l_2 = \max(0, 11-10-1) = 0$

$L = 0$

Greedy algo template: Order the intervals/jobs

→ schedule them right one after the other



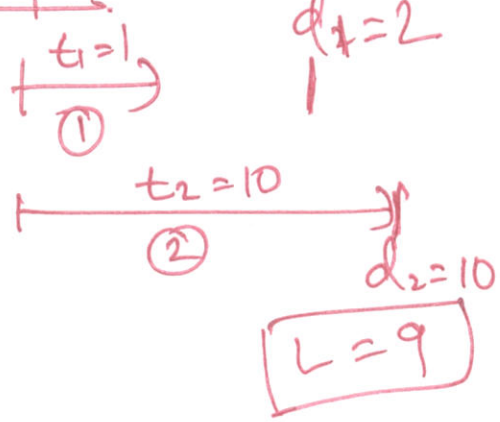
Q: What kind of orderings to consider?

A1: Increasing order of t_i ✗

A2: ——— d_i ✓

A3: ——— slack $= d_i - t_i$ ✗

Example:



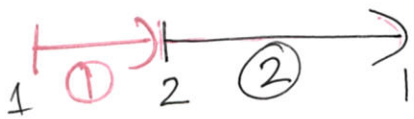
$L=9$

Schedule 1 (increasing order of d_i)

$\ell_1 = \max(0, 12-2-1) = 9$

$\ell_2 = \max(0, 11-10-1) = 0$

Schedule 2 (increasing order of d_i)



$\ell_1 = \max(0, 2-2-1) = 0$

$\ell_2 = \max(0, 12-10-1) = 1$

$L=1$

Deadline first scheduling — $O(n \log n)$

0. Sort the jobs by deadline ($\Rightarrow d_1 \le d_2 \le \dots \le d_n$) ← $O(n \log n)$
1. $f = 1$ ← $O(1)$ — break ties arbitrarily
2. for $i = 1 \dots n$ ← n iterations
   - $s(i) = f$
   - $f(i) = s(i) + t_i$
   - $f = f(i)$

   $O(n)$
3. Return the schedule $i \mapsto [s(i), f(i))$ ← $O(n)$ — let the output be S

Overall $= O(n \log n) + O(n) = O(n \log n)$

S: output of greedy algo

O: be an optimal solution (among all valid schedules, O has the min max lateness)

Dream: $S = O$ [Problem: multiple optimal solutions possible]

THM: $L(S) = L(O)$.