Scheduling to minimize lateness

I/p: n intervals/jobs/tasks

$i \mapsto (d_i, t_i)$

d_i: deadline, t_i: duration

global start time: s

O/p: Schedule : $i \mapsto s(i)$ $(\geq s)$ — for EVERY $1 \leq i \leq n$

$\Rightarrow$ i gets scheduled $[s(i), s(i)+t_i)$ — def $f(i)$

s.t no $i \neq j$ are in conflict.

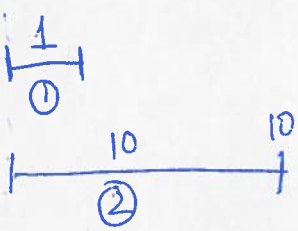
Goal: Min max lateness (min L)

($l_i = \max(f(i) - d_i, 0)$)

lateness, max lateness $L = \max_i l_i$)

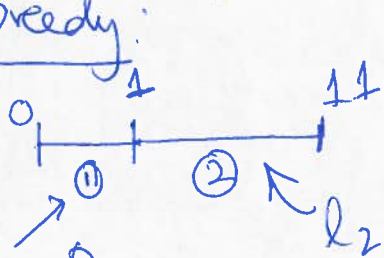
Greedy algos: Define a number for each i & sort by that number.

Option 1: t_i (shortest first)

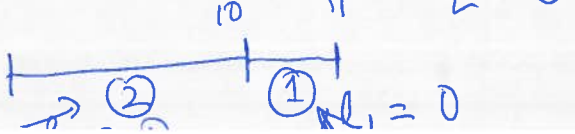


100
1

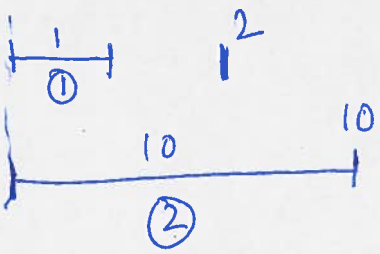
Greedy:



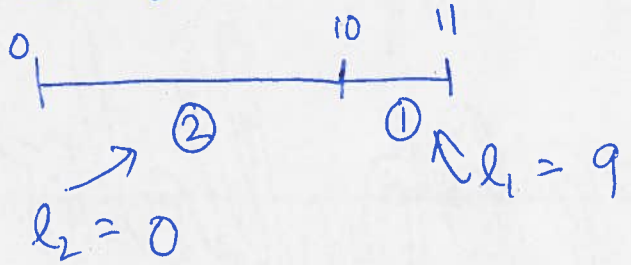
OPT:



Option 2: slack, $d_i - t_i$

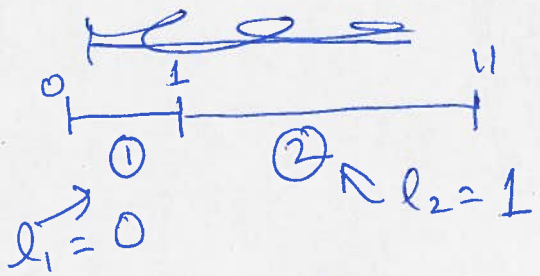


Greedy:



$L = 9$

OPT:



$L = 1$

Option 3: Sort by deadline

Sound fishy!

Algo:

0. Sort all jobs by deadlines
1. Assume $d_1 \le d_2 \le \dots \le d_n$
2. $f \leftarrow s$
3. for $i = 1 \dots n$

   ~~f(i) ←~~

   $s(i) \leftarrow f$

   $f(i) \leftarrow s(i) + t_i$

   $f \leftarrow f(i)$
4. Return the schedule, $i \mapsto [s(i), f(i))$

   $\forall\ 1 \le i \le n$