Scheduling jobs with the minimum delay

Set of jobs = $\{ (t(i), d(i)) \mid 1 \le i \le n \}$

$\hookrightarrow$ deadlines

$\hookrightarrow$ time for the job

A schedule is given by $(s(i), f(i))$ for each job i where $f(i) = s(i) + t(i)$

The delay is given by $\ell(i) = \begin{cases} 0 & \text{if } f(i) \le d(i) \\ f(i) - d(i) & \text{o/w} \end{cases}$

$$\boxed{L = \max_i \{\ell(i)\}}$$

$\longrightarrow$ find a schedule S that minimizes L

Example: (2, 4), (3, 6), (1, 2)

S = (1, 2), (2, 4), (3, 6) $L = 0$

S' = (2, 4), (3, 6), (1, 2) $L = 4$

Greedy 1: Choose the job with the shortest t_i

S = (1, 8), (2, 4), (3, 2) $L = 4$

S' = (3, 2), (2, 4), (1, 8) $L = 1$

Greedy 2: Choose the job i with the smallest $d(i) - t(i)$

S = (4, 4), (1, 2) $L = 3$

S' = (1, 2), (4, 4) $L = 1$

Greedy choice: Choose the job with the smallest $d(i)$

* $\exists$ optimal schedule that has no idle time

making jobs start earlier cannot increase delay

no gap between jobs

* An inversion is a pair of jobs (i, j) s.t i is scheduled before j and

$d(j) < d(i)$

(earliest deadline first creates a schedule that has no inversions)

* Every schedule that has no idle time and no inversions have the same max delay (L)

In every schedule that has no idle time & no inversions, the jobs with the same deadline occur consecutively

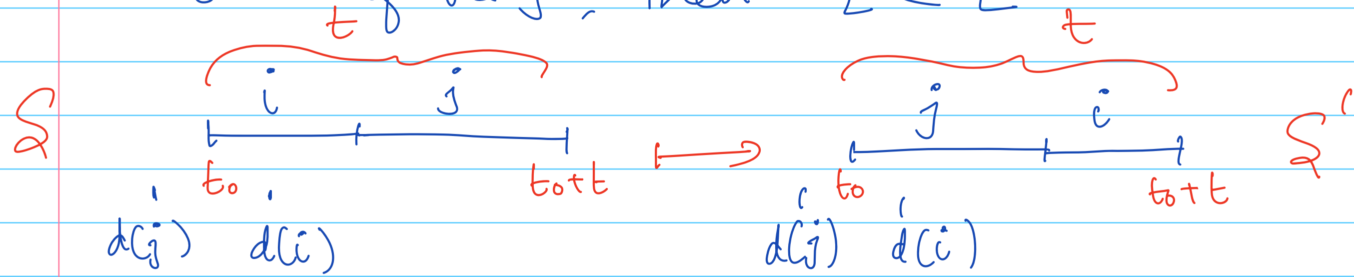
& permuting them does not change the delay

* If a schedule S has an inversion, then $\exists$ two jobs i, j such i is scheduled directly before j & $d(i) > d(j)$ → inversion (i,j) occur consecutively

i' i_1 i_2 . . . . . i_k j'

$d(i') <$ $<$ ~ $>$ $d(j')$ $\quad d(i') > d(j')$

* Let S have consecutive jobs (i,j) forming an inversion. Let S' be obtained by swapping the order of i & j, then $L' \leq L$



- delays of all jobs apart from i & j remain the same
- $f'(i) = f(j)$
- delay of j can only reduce
- $l'(i) = f'(i) - d(i) = f(j) - d(i) < f(j) - d(j)$

$\qquad = l(j)$

$L \geq l(j) \geq l'(i) \Rightarrow L' \leq L$

* Keep swapping adjacent inversions to obtain schedule with no idle time & inversions

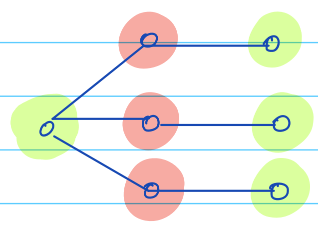
Vertex Cover

Graph $G(V,E)$: Goal: Find the smallest set $V' \subseteq V$ s.t

$\forall (u,v) \in E$, $u \in V'$ or $v \in V'$

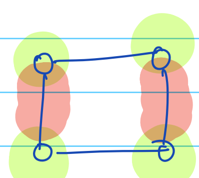
– NP-hard: no known poly-time algorithm

Greedy 1: Find v with largest degree and add to V'. Remove e with endpoints in v



OPT = 3
Greedy 1 = 4

Greedy 2: For each $e \in G$, add both endpoints to V'; Remove all edges incident to either of the endpoints



OPT = 2
Greedy 2 = 4

which of these is a better greedy algorithm?

Theorem: Greedy 2 always outputs V' s.t

$$|V'| \le 2 \cdot |OPT|$$

Proof: V' obtained by G2 is a vertex cov

for each considered by G2, one of its endpoints must be present in OPT $\Rightarrow$ $OPT \ge \frac{|V'|}{2}$

Theorem: Greedy 1 always outputs V' s.t

$$|V'| = |OPT| \cdot O(\log n)$$