Apr 9

Weighted Interval Scheduling

Simplified problem: Instead of outputting an optimal solution $\mathcal{O}$, output $v(\mathcal{O}) = \sum_{i \in \mathcal{O}} v(i)$

$$OPT(J) \overset{\text{def}}{=} \text{value of an optimal solution for } [J] \begin{cases} (s_1, f_1, v_1) \\ (s_2, f_2, v_2) \\ \vdots \\ (s_J, f_J, v_J) \end{cases}$$

$1 \le J \le n$

ASSUME: $f_1 \le f_2 \le \dots \le f_n$

Goal: Compute $OPT(n)$

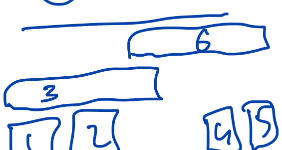
Def: $\mathcal{O}_J$ be an optimal solution for $[J]$

$$v(\mathcal{O}_J) = OPT(J)$$

Case 1: $J \notin \mathcal{O}_J$ Claim 1: $\mathcal{O}_J$ is an optimal solution for $[J-1]$

$$\implies OPT(J) = OPT(J-1) \quad (1)$$

Case 2: $J \in \mathcal{O}_J$ Claim 2: $\mathcal{O}_J \setminus \{J\}$ is an optimal solution for $[p(J)]$



$p(6) = 2$

Def: $p(J)$ = largest value i s.t. i & J do not conflict

$= 0$ if no such i $\exists$

$$\Longrightarrow OPT(j) = OPT(p(j)) + v_j \quad ②$$

Combining ① & ② $\Rightarrow OPT(j) = \max \{ OPT(j-1), \; v_j + OPT(p(j)) \}$