→ have access to $p(1), \ldots, p(n)$
→ Assumed $f_1 \leq f_2 \leq \ldots \leq f_n$

Nov 12

$\Rightarrow M[0 \ldots n]$

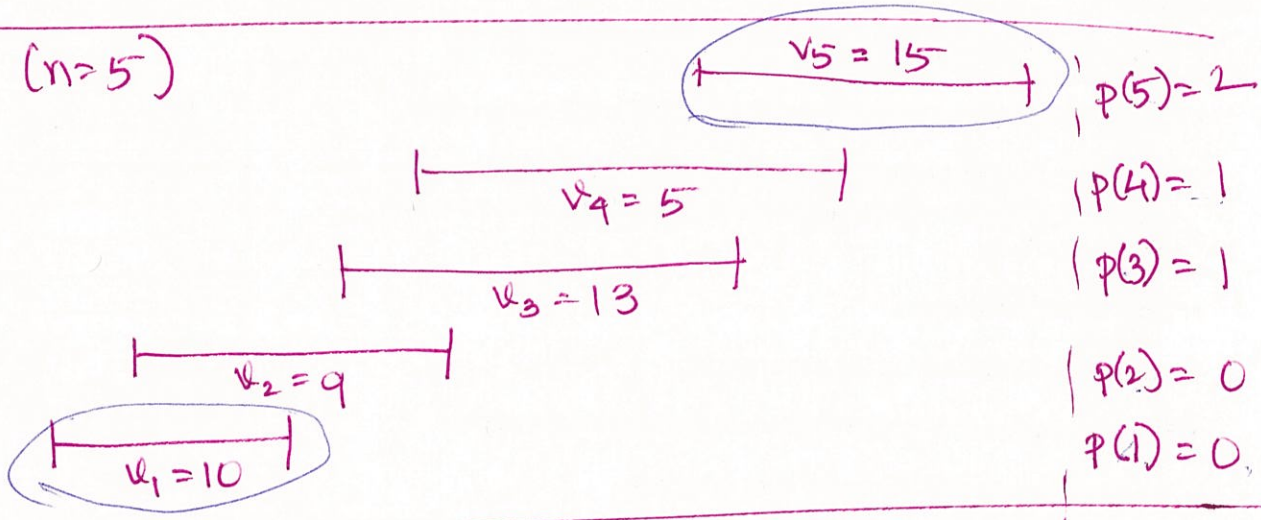
(•) $M[0] \leftarrow 0$

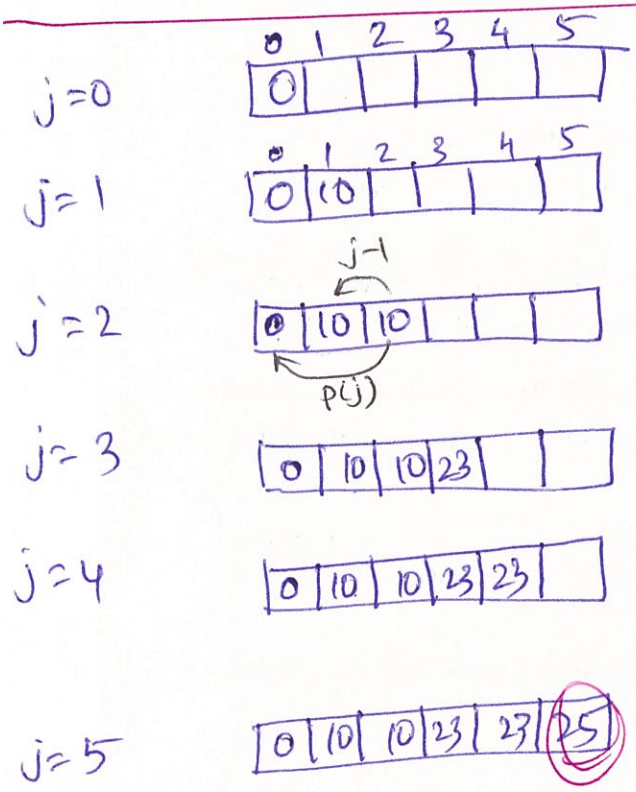
(•) for $j = 1 \ldots n$

$M[j] \leftarrow \max \{ v_j + M[p(j)], M[j-1] \}$

(•) return $M[n]$

Example ($n = 5$)





$M[0] \leftarrow 0$

$p(1) = 1$

$M[1] \leftarrow \max \{ v_1 + M[0], M[0] \}$
$= \max \{ 10 + 0, 0 \} = 10$

$M[2] \leftarrow \max \{ v_2 + M[0], M[1] \}$
$= \max \{ 9 + 0, 10 \} = 10$

$M[3] \leftarrow \max \{ v_3 + M[1], M[2] \}$
$= \max \{ 13 + 10, 10 \} = 23$

$M[4] \leftarrow \max \{ v_4 + M[1], M[3] \}$
$= \max \{ 5 + 10, 23 \} = 23$

$M[5] \leftarrow \max \{ v_5 + M[2], M[4] \}$
$= \max \{ 15 + 10, 23 \} = 25$

→ $OPT(5) = 25$

Q: What is O_5?

Recall: $j \in \mathcal{O}_j$ if $v_j + OPT(p(j)) > OPT(j-1)$

Compute optimal schedule for n=5 example $OPT(0), \ldots, OPT(5)$

n=5: $5 \overset{?}{\in} \mathcal{O}_5$ $15 + 10 \overset{?}{>} 23$ ✓

$\Rightarrow 5 \in \mathcal{O}_5$

$p(5) = 2$ Considering $\mathcal{O}_5 \setminus \{5\} = \mathcal{O}_2$ for [2]

$2 \overset{?}{\in} \mathcal{O}_2$ $9 + 0 > 10$ ✗ $\Rightarrow 2 \notin \mathcal{O}_2$

$p(2) = 0$

Consider $\mathcal{O}_2 = \mathcal{O}_1 \Rightarrow 10 + 0 > 0$ ✓ $\Rightarrow 1 \in \mathcal{O}_1$

$\Rightarrow$ $\{1,5\}$ is an opt. solution

```
MSchedule (n; M, p)
  If n=0 then return ϕ
  If v_n + M[p(n)] > M[n-1]
      return {n} ∪ MSchedule(p(n); M, p)
  else
      return MSchedule(n-1; M, p)
```

SUBSET SUM problem

Ex: $n = 3$ $w_1 = 1$, $w_2 = 3$, $w_3 = 3$

Output a subset of these 3 numbers s.t their sum $\leq W$ & max such sum.

(i) $W = 7 \Rightarrow$ opt soln $= \{1,2,3\}$

(ii) $W = 6 \Rightarrow$ opt soln $= \{2,3\}$

(iii) $W = 5 \Rightarrow$ opt soln $= \{1,2\}$ or $\{1,3\}$ → In general cannot have sum of values in OPT $= W$.

Input: n integers: $w_1, \ldots, w_n$; $w_i > 0$ Budget: $W \geq 0$

Output: A subset $S \subseteq [n]$ s.t.

(i) $\sum_{i \in S} w_i \leq W$ (ii) max $w(S) = \sum_{i \in S} w_i$

Simpler Q: max $|S|$ instead of $w(S)$ in (ii)

Greedy algo: Sort the w_i's in increasing order
pick as many as possible w/o exceeding W

Ex: Prove this is optimal (simpler Q): Greedy stays ahead

Our problem: $w(S)$

↳ Greedy algo doesn't work: counter ex: $W = 6$
Greedy will pick $\{1, 3\}$ with ~~total~~ weight $= 4$
but $OPT = 6$

NOTE: No known greedy algo.