SUBSET SUM PROBLEM

Nov 20

Input: n integers $w_1, \dots, w_n$; $w_i > 0$

~~Bound~~ Budget $W \geq 0$

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

(i) $\sum_{i \in S} w_i \leq W$

(ii) maximizes $w(S) \overset{\text{def}}{=} \sum_{i \in S} w_i$

Example: $n = 3$. $w_1 = 1, w_2 = 3, w_3 = 3$

(i) $W = 7 \Rightarrow$ solution: $\{1, 2, 3\}$

(ii) $W = 6 \Rightarrow$ solution: $\{2, 3\}$

(iii) $W = 5 \Rightarrow$ solution: $\{1, 2\}$ or $\{1, 3\}$

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

Q1: Greedy algo to solve for max $|S|$?

A1: Sort by increasing w_i's and pick as many as you can without exceeding the budget W.

Ex: Prove this is optimal (using "greedy stays ahead")

Next: Original problem of max $w(S)$

Try 1: Use the greedy algo as above.

↳ **Counter-example:** $w_1 = 1$, $w_2 = w_3 = 3$, $W = 6$

Greedy outputs $\{1, 2\}$ or $\{1, 3\}$ but optimal $= \{2, 3\}$

NOTE: No known greedy algorithms for this problem

GOAL: Design a dynamic programming algo. Instead of trying to compute S compute $w(S)$ for some optimal S

Let O_j be an optimal solution for $w_1, \dots, w_j$

$OPT(j) = w(O_j)$

Case 1: $j \notin O_j \Rightarrow OPT(j) = OPT(j-1)$

Claim: O_j is an optimal solution for $1, \dots, j-1$

Pf (idea): Let's say O' is a better than O_j for ______ $\Rightarrow$

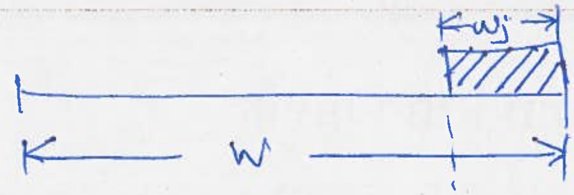
______ $1, \dots, j$

Case 2: $j \in O_j$

Q: What can we say about $O_j \setminus \{j\}$

Hope: $O_j \setminus \{j\}$ is an optimal solution for $1 .., j'$ $\quad j' < j$

If so, $OPT(j) = w_j + OPT(j')$



new budget $= W - w_j$

Solution: Keep track of budget and j

$OPT(j, B) \doteq$ weight of an optimal solution on $w_1, .., w_j$ and budget B.

(Back to $j \in O_j$)

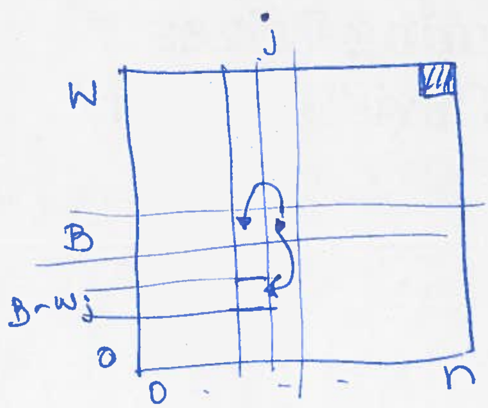
$OPT(j, B) = w_j + OPT(j-1, B - w_j)$ $\quad$ Need: $w_j \le B$

Case 1: $j \notin O_j \Rightarrow OPT(j, B) = OPT(j-1, B)$

Overall recursion:

If $w_j > W \Rightarrow OPT(j, W) = OPT(j-1, W)$

else $OPT(j, W) = \max\{\underbrace{w_j + OPT(j-1, W - w_j)}_{j \in O_j}, \underbrace{OPT(j-1, \text{}}_{j \notin O_j}$



$0 \le j \le n$

$0 \le B \le$

$M[B][j] = OPT(j, B)$

Q1: How many sub-problems?

$(W+1)(n+1)$

Q2: What entry do we want?

$M[W][n] = OPT(n, W)$

Q3: Ordering? $\quad OPT(j, B) \rightarrow$ go column by column