Nov 17

SUBSET SUM problem

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

Budget: $W \geq 0$

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

(i) $w(S) \overset{def}{=} \sum_{i \in S} w_i \leq W$ (ii) maximize $w(S)$ {among all such valid S}

Simplified version 2.0: Compute max $w(S)$ $\left\{\begin{matrix}\text{over all } S \\ \text{satisfying (i)}\end{matrix}\right\}$

Another simplified: max $|S|$ instead of $w(S)$

Greedy algo: sort w_i's in ~~increasing~~ (non-decreasing) & pick as many in order as possible ~~while~~ w/o exceeding W

Ex! Greedy algo optimize $|S|$ Hint: Greedy stays ahead proof.

Greedy algo doesn't work for max $w(S)$

Ex! $w_1 = 1,\ w_2 = 3,\ w_3 = 3$ $\quad W = 6$

greedy opt

Note: no known greedy algo

Dynamic program

Attempt 1: O_j be optimal solution $w_1, \dots, w_j$

$OPT(O_j) = w(O_j)$

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

(Ex) Claim: O_j is still optimal for $w_1, \dots, w_{j-1}$

Case 2: $j \in O_j$

Q! What can we say for $O_j \setminus \{j\}$

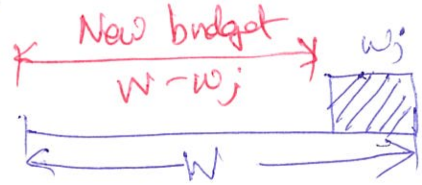
Hope: Somehow argue $O_j \setminus \{j\}$ is optimal $w_1, \dots, w_{j'}$ for some $j' < j$

$\Longrightarrow OPT(j) = w_j + OPT(j')$

Q! What is wrong with the above hope?

→ What is the actual sub-problem when we pick j?

$w_1, \dots, w_j ; W \longrightarrow w_1, \dots, w_{j-1} ; W - w_j$



Define subproblems $w_1, \dots, w_j ; B$

Def: $OPT(B, j)$ = wt. of the optimal solution for input $w_1, \dots, w_j ; B$

$0 \le j \le n$

$0 \le B \le W$

Goal:

(*) $\boxed{\begin{array}{l} \text{If } w_j > B \text{ then } \quad OPT(B, j) = OPT(B, j-1) \\ \text{else } \quad OPT(B, j) = \max\{ \underbrace{w_j + OPT(B - w_j, j-1)}_{\text{Case 2}}, \underbrace{OPT(B, j-1)}_{\text{Case 1}} \} \end{array}}$

Case 1: $j \notin \mathcal{O}(j; B)$ Assume: $w_j \le B$

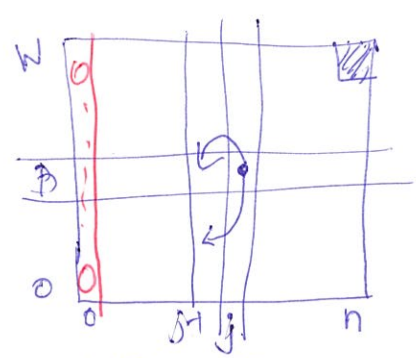
$\Rightarrow OPT(B, j) = OPT(B, j-1)$

Case 2: $j \in \mathcal{O}(j; B)$

$OPT(B, j) = w_j + OPT(B - w_j, j-1)$

If $w_j > B \Rightarrow j \notin \mathcal{O}(j; B) \Rightarrow OPT(B, j) = OPT(B, j-1)$

$OPT(B, 0) = 0 \quad \forall B \quad 0 \le B \le W$



Q1) Given input $w_1, \dots, w_n ; W$

If $M[B, j]$ is computed correctly

$\forall \; 0 \le j \le n, \quad 0 \le B \le W$

the output is $M[W, n]$

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

Q2) Initialize $OPT(B, 0) = 0 \; \forall B$

(Q3) How many sub-problems? $(n+1)(W+1) = O(nW)$ Assume this

$\hookrightarrow poly(n)$ if $W \le poly(n)$

(Q4) Recurrence between sub-problems? → (*)

(Q5) Ordering among sub-problems?

→ Go in order of columns $M[:,0]$, $M[:,1]$, ... $M[:,j-1]$, $M[:,j]$, ...

SubsetSum$(w_1, \dots, w_n; W)$

0. Allocate $(W+1) \times (n+1)$ matrix M $\}$ $O(nW)$

1. $M[B,0] \leftarrow 0 \quad \forall\ 0 \le B \le W$ $\}$ $O(W)$

2. for $j = 1 .. n$ ~~Order~~

   for $B = 0 .. W$

   if $w_j > B$

   $M[B,j] \leftarrow M[B,j-1]$

   else

   $M[B,j] \leftarrow \max\{ w_j + M[B-w_j, j-1], M[B,j-1]\}$

   $O(1)$ (inner if/else); $O(nW)$ (loop)

3. return $M[W,n]$ ← $O(1)$

Ex: Prove by induction that $M[B,j] = OPT(B,j)$

Overall runtime $= O(nW)$ pseudopoly ⇒ poly if $W \le poly(n)$

In general input size $N = n + \log W$

If $W = 2^n \Rightarrow N = \Theta(n)$ but runtime $O(n \cdot 2^n)$