Apr 27

Subset-Sum $(w_1, \ldots, w_n; W)$

0. Allocate an $(W+1) \times (n+1)$ matrix } $O(Wn)$
1. $M[B,0] \leftarrow 0 \quad \forall\ 0 \le B \le W$ ← $O(W)$
2. for $j = 1 \cdots n$ {
   for $B = 0 \cdots W$ $O(nW)$
   if $w_j > B$ $O(1)$
   $M[B,j] \leftarrow M[B,j-1]$
   else $M[B,j] \leftarrow \max\{w_j + M[B-w_j, j-1], M[B,j-1]\}$

3. return $M[W,n]$

Runtime: $O(nW)$ ← pseudopolynomial
$\Rightarrow$ polynomial if $W = \text{poly}(n)$

Input size: $n + \log W$

$W = 2^n \rightarrow$ input size: $\Theta(n)$; Runtime: $O(n \cdot 2^n)$

Note: 1. $O(W)$ space if we are only interested in the value of subset.

2. $\Omega(nW)$ space if we want to know/output the actual subset

$\Rightarrow$ Use the same or similar recursive solution that we saw in weighted interval scheduling problem.

$\underline{n=3}$ $\quad w_1 = 1 \quad w_2 = 2 \quad w_3 = 3$

$W = 3$

M:

3	0			
2	0	1		
1	0	1	1	
0	0	0	0	0
	0	1	2	3

$B=1 \quad w_1 = 1 \mid w_j \leq B$

$1 \leq 1$

$M[1,1] \leftarrow \max\{1+M[1-1,0],\ M[1,0]\}$

$= \max\{1+0,\ 0\} = 1$

$M[1,2]:\ B: 1,\ w_2 = 2$

$w_2 > B$

$M[1,2] \leftarrow M[1,1]$

$= 1$

$M[2,1]:\ B: 2,\ w_1: 1$

$w_1 = 1 \leq B = 2$

$M[2,1] \leftarrow \max\{1+M[2-1,0],\ M[2,0]\}$

$= \max\{1+0,\ 0\} = 1$

Knapsack problem

Shortest Path problem

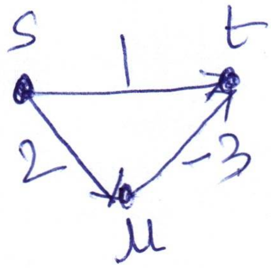
Input: (•) Directed graph $G = (V, E)$

$\forall e \in E$, c_e ($c_e < 0$ allowed)

(•) $t \in V$

Output: $\forall s \in V$ output shortest s-t path.

Attempt 1 Use Dijkstra's algo. on each $s \in V$

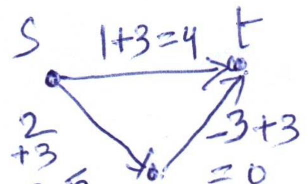


Shortest path: s, u, t ; cost: 0

Dijkstra's will pick: s, t ; cost: 1

Doesn't work ✗ $1 > 0$

Attempt 2: Add 4 to all edges to make costs non-negative.



Dijkstra's will output: s, t

$\leftrightarrow$ correct for the modified graph

$\rightarrow$ not " " " original "

No known greedy / divide and conquer algo.

Assume: We are interested only in the cost of the shortest s-t path.

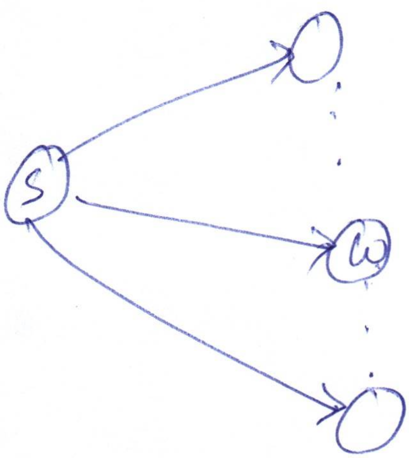
Goal: Design a dynamic program.

Attempt 3: $OPT(s)$ = cost of shortest s-t path $\forall s \in V$.

(•) How many subproblems?

$\# OPT(s)$? n ✓

(•) Recurrence / recursive defn of subproblems:



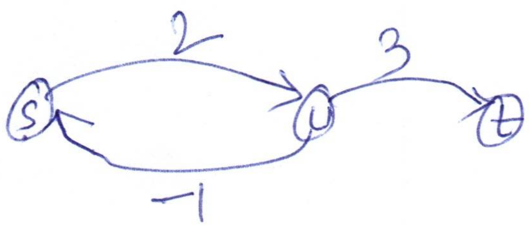
if (s,w) in in shortest s-t path:

$$OPT(s) = c_{s,w} + OPT(w)$$

In general,

$$OPT(s) = \min_{w:\,(s,w)\in E} \{c_{s,w} + OPT(w)\}$$

(•)



$$OPT(s) = 2 + OPT(u)$$

$$OPT(u) = \min\{3 + OPT(t),\ -1 + OPT(s)\}$$

ISSUE: $OPT(s)$ depends on $OPT(u)$

$OPT(u)$ " " $OPT(s)$