Computer Science
Spring 2021 Lecture 15:
Dynamic Programming:
Subset Sum

2

The Subset Sum Problem

T: target

Problem Statement: Given a set S of n positive integers, as well as a positive integer T, determine if there is a subset of S that sums to exactly T.
Example 1: $S = \{2, 3, 4\}$, $T = 6$, answer is "yes"
Example 2: $S = \{2, 3, 5\}$, $T = 6$, answer is "no"

Subset Sum: recursive solution

3

As with any dynamic programming problem

- Try a recursive approach first
- Find a tautology, then list decisions

Sub(n) // does a boolean subset of S[1..n] add to T?

if 0 == n? only if T is zero?

//else

if_nth_not_used = Sub(n-1)

if_nth_used = ~~S[n] + S(n-1)~~?

need ⊆ S[1..n] to add to T-S[n]

Recursive solution, attempt two

4

Sub(n,T) : "does a subset of $S[1 \ldots n]$ add to T?

// Tautology: if yes, $S[n]$ is used or it is not

// if_no = Sub($n-1$, T)

// if_yes = Sub(n-1, T - S[n]) ← assumes $T \geq S[n]$

// Now the code:

if 0 == T return true; // $\sum_{x \in \emptyset} x = 0$

else if 0 == n return false;

else return Sub(n-1, T) ||

(T ≥ S[n] && Sub(n-1, T-S[n]))

5 ## Subset Sum: iterative solution

does a subset of S[1..i] add to j?

SubsetSum(i, j) // recursive for reference

if $0 = j$ **then**
  return true
else if $0 = i$ **then**
  return false
else
  return SubsetSum($i-1, j$) OR
  ($\cancel{T}^{j} - S[\cancel{n}^{i}] \geq 0$ and SubsetSum($\cancel{n}^{i} - 1, \cancel{T}^{j} - S[\cancel{n}^{i}]$))

Declare Sub[0...n, 0...T]
for i = 0 to n Sub[i,0] = true
for j = 1 to T Sub[0,j] = false
for i = 1 to n
  for j = 1 to T
    Sub[i,j] = Sub[i-1,j] || (j ≥ S[i] && Sub(i-1, j-S[i]))

6 ## Subset Sum: Visualization

Example 1: $S = \{2, 3, 4\}$, $T = 6$.

		0	1	2	3	4	5	6
i=0	$\{\}$	T	F	F	F	F	F	F
i=1	$\{2\}$	T	F	T	F	F	F	F
i=2	$\{2, 3\}$	T	F	T	T	T	T	F
	$\{2, 3, 4\}$	T	F	T	T	T	T	T

Find subset
(from end of lec)

i = ~~3~~ ~~2~~ ~~1~~ 0 j = ~~6~~ ~~2~~ 0

output: 4
2

7

Subset Sum: Running Time

```
SubsetSum(S[1...n], T) // iterative
  for i = 0...n do
    SUB[i, 0] = true
  for j = 1...T do
    SUB[0, j] = false
  for i = 1...n do
    for j = 1...T do
      Fill in SUB[i, j] in O(1)
  return SUB[n, T]
```

$\}\ \Theta(n)$

$\}\ \Theta(T)$

$\}\ \Theta(nT)$

- What is the running time of Subset Sum?

$\Theta(nT)$

7

Subset Sum: Running Time

$\Theta(nT)$: pseudo-polynomial

```
SubsetSum(S[1...n], T) // iterative
  for i = 0...n do
    SUB[i, 0] = true
  for j = 1...T do
    SUB[0, j] = false
  for i = 1...n do
    for j = 1...T do
      Fill in SUB[i, j] in O(1)
  return SUB[n, T]
```

size of input: n, $\log_2 T$

$\Theta(n^5 (\log T)^{100})$ would be polynomial

- Suppose we double the size of S, but leave T alone. Will your algorithm scale well? ✓
- Suppose we double the **size** of T, but leave S alone. Will your algorithm scale well?

8

Subset Sum: Find the Subset

```
SubsetSum(S[1...n], T) // iterative
  for i = 0...n do
    SUB[i, 0] = true
  for j = 1...T do
    SUB[0, j] = false
  for i = 1...n do
    for j = 1...T do
      Fill in SUB[i, j] in O(1)
  if SUB[n, T] is true then
      i ← n , j ← T
      while i > 0 :   // SUB[i,j] true. use S[i]?
          if  S[i] ≤ j  and  Sub[i-1, j-S[i]]
                output  S[i]
                j ← j - S[i]
          i--
```