CompSci 161
Winter 2023 Lecture 12:
Dynamic Programming III:
Subset Sum
The Subset Sum Problem

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

As with any dynamic programming problem
  ▶ Try a recursive approach first
  ▶ Find a tautology, then list decisions

\[ \text{Sub}(n) : \text{"is there a subset of } S[1...n] \text{ that adds to } T? \]

use \( s[n] \) or don't  \( \Rightarrow \text{Sub}(n-1) \)
Recursive solution, attempt two

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]) // unless $S[n] > T$

base cases? $T=0 \rightarrow$ yes.
// Now the code:
T>0 and n=0 : no.

if $T=0$ return true
elif $n=0$ return false
else return Sub(n-1, T) or $(T \geq S[n]$ and $Sub(n-1, T-S[n])$)
Subset Sum: iterative solution

SubsetSum\((i,j)\) // recursive for reference

\[
\begin{align*}
\text{if } 0 &= j \text{ then} \\
&\quad \text{return } \text{true} \\
\text{else if } 0 &= i \text{ then} \\
&\quad \text{return } \text{false} \\
\text{else} \\
&\quad \text{return } \text{SubsetSum}\((i-1,j)\) \text{ OR} \\
&\quad \quad \quad \text{(} j - S[i] \geq 0 \text{ and SubsetSum}\((i-1,j - S[i])\)\)}
\end{align*}
\]

\[
\begin{align*}
\text{for } i &= 0 \ldots n \\
&\quad \text{Sub}[i,0] = \text{true} \\
\text{for } j &= 1 \ldots T \\
&\quad \text{Sub}[0,j] = \text{false} \\
\text{for } i &= 1 \ldots n \\
&\quad \text{for } j &= 1 \ldots T \\
&\quad \text{fill in } \text{Sub}[i,j] \text{ as per above}\}
\end{align*}
\]
Subset Sum: Visualization

Example 1: \( S = \{2, 3, 4\}, \ T = 6. \)
Subset Sum: Running Time

\texttt{SubsetSum(S[1...n], T) // iterative}
\begin{verbatim}
  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]
\end{verbatim}

What is the running time of Subset Sum?
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]

▶ 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?
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

\[ i \leftarrow n, j \leftarrow T \] // Sub[i, j] true

while i > 0:
    if ! SUB[i-1, j]
        Output \[ S[i] \]
        \[ j = j - S[i] \]
    i--