Knapsack Problem

**Input**: an integer bound $W > 0$

- a set of $n$ items, each with an integer weight $w_i > 0$
- a value $v_i > 0$ for each item $i$

**Output**: a subset $S$ of items that

maximizes $\sum_{i \in S} v_i$  s.t. $\sum_{i \in S} w_i \leq W$. 

Motivation: you have budget $W$, and want to buy a subset of items of maximum total value.
Knapsack Problem

**Input:** an integer bound $W > 0$

a set of $n$ items, each with an integer weight $w_i > 0$

a value $v_i > 0$ for each item $i$

**Output:** a subset $S$ of items that

maximizes $\sum_{i \in S} v_i$ \quad s.t. $\sum_{i \in S} w_i \leq W$.

- Motivation: you have budget $W$, and want to buy a subset of items of maximum total value
DP for Knapsack Problem

- $opt[i, W']$: the optimum value when budget is $W'$ and items are \{1, 2, 3, \ldots , i\}.
- If $i = 0$, $opt[i, W'] = 0$ for every $W' = 0, 1, 2, \ldots , W$.

\[
opt[i, W'] = \begin{cases} 
  & i = 0 \\
  & i > 0, w_i > W' \\
  & i > 0, w_i \leq W'
\end{cases}
\]
DP for Knapsack Problem

- $opt[i, W']$: the optimum value when budget is $W'$ and items are $\{1, 2, 3, \ldots, i\}$.
- If $i = 0$, $opt[i, W'] = 0$ for every $W' = 0, 1, 2, \ldots, W$.

$$
opt[i, W'] = \begin{cases} 
0 & i = 0 \\
& i > 0, w_i > W' \\
& i > 0, w_i \leq W' 
\end{cases}
$$
DP for Knapsack Problem

- \( opt[i, W'] \): the optimum value when budget is \( W' \) and items are \( \{1, 2, 3, \cdots, i\} \).
- If \( i = 0 \), \( opt[i, W'] = 0 \) for every \( W' = 0, 1, 2, \cdots, W \).

\[
\begin{align*}
\text{opt}[i, W'] &= \begin{cases} 
0 & i = 0 \\
\text{opt}[i - 1, W'] & i > 0, w_i > W' \\
\text{opt}[i - 1, W'] & i > 0, w_i \leq W'
\end{cases}
\end{align*}
\]
DP for Knapsack Problem

- \( \text{opt}[i, W'] \): the optimum value when budget is \( W' \) and items are \( \{1, 2, 3, \ldots, i\} \).
- If \( i = 0 \), \( \text{opt}[i, W'] = 0 \) for every \( W' = 0, 1, 2, \ldots, W \).

\[
\text{opt}[i, W'] = \begin{cases} 
0 & \text{if } i = 0 \\
\text{opt}[i - 1, W'] & \text{if } i > 0, w_i > W' \\
\max \left\{ \text{opt}[i - 1, W'], \text{opt}[i - 1, W' - w_i] + v_i \right\} & \text{if } i > 0, w_i \leq W'
\end{cases}
\]
Exercise: Items with 3 Parameters

**Input:** integer bounds $W > 0$, $Z > 0$,
a set of $n$ items, each with an integer weight $w_i > 0$
a size $z_i > 0$ for each item $i$
a value $v_i > 0$ for each item $i$

**Output:** a subset $S$ of items that

\[
\text{maximizes } \sum_{i \in S} v_i \quad \text{s.t.} \\
\sum_{i \in S} w_i \leq W \text{ and } \sum_{i \in S} z_i \leq Z
\]
Outline

1 Weighted Interval Scheduling
2 Subset Sum Problem
3 Knapsack Problem
4 Longest Common Subsequence
   • Longest Common Subsequence in Linear Space
5 Shortest Paths in Directed Acyclic Graphs
6 Matrix Chain Multiplication
7 Optimum Binary Search Tree
8 Summary
Subsequence

- $A = bacdca$
- $C = adca$
Subsequence

- $A = bacdca$
- $C = adca$
- $C$ is a subsequence of $A$
Subsequence

- \( A = bacdca \)
- \( C = adca \)
- \( C \) is a subsequence of \( A \)

**Def.** Given two sequences \( A[1 .. n] \) and \( C[1 .. t] \) of letters, \( C \) is called a subsequence of \( A \) if there exists integers \( 1 \leq i_1 < i_2 < i_3 < \ldots < i_t \leq n \) such that \( A[i_j] = C[j] \) for every \( j = 1, 2, 3, \ldots, t \).
Subsequence

- $A = bacdca$
- $C = adca$
- $C$ is a subsequence of $A$

**Def.** Given two sequences $A[1 \ldots n]$ and $C[1 \ldots t]$ of letters, $C$ is called a **subsequence** of $A$ if there exists integers $1 \leq i_1 < i_2 < i_3 < \ldots < i_t \leq n$ such that $A[i_j] = C[j]$ for every $j = 1, 2, 3, \ldots, t$.

Exercise: how to check if sequence $C$ is a subsequence of $A$?
Common subsequence

**Def.** Given two sequences $A[1 \ldots n]$ and $B[1 \ldots m]$ of letters, $C$ is called a **common subsequence** of $A$ and $B$ if $C$ is a subsequence of $A$ and also a subsequence of $B$. 
Def. Given two sequences $A[1 \ldots n]$ and $B[1 \ldots m]$ of letters, $C$ is called a common subsequence of $A$ and $B$ if $C$ is a subsequence of $A$ and also a subsequence of $B$.

Example: $A = adecaf$ and $B = caefcad$
Def. Given two sequences $A[1 .. n]$ and $B[1 .. m]$ of letters, $C$ is called a common subsequence of $A$ and $B$ if $C$ is a subsequence of $A$ and also a subsequence of $B$.

- Example: $A = adecadf$ and $B = caefcad$
- Common subsequence: $C = adcaf$
Common subsequence

**Def.** Given two sequences $A[1 \ldots n]$ and $B[1 \ldots m]$ of letters, $C$ is called a common subsequence of $A$ and $B$ if $C$ is a subsequence of $A$ and also a subsequence of $B$.

- Example: $A = adecadf$ and $B = caefcad$
- Common subsequence: $C = adcaf$ ?
- Common subsequence: $C = aead$ ?
Common subsequence

**Def.** Given two sequences $A[1 .. n]$ and $B[1 .. m]$ of letters, $C$ is called a **common subsequence** of $A$ and $B$ if $C$ is a subsequence of $A$ and also a subsequence of $B$.

- Example: $A = adecadf$ and $B = caefcad$
  - Common subsequence: $C = adcaf$  
  - Common subsequence: $C = aead$  
  - Common subsequence: $C = acad$
Edit distance with two operations (insertions and deletions)

**Def.** Given two sequences $A[1..n]$ and $B[1..m]$ of letters, $d(A, B)$ is called a edit distance with insert and delete operations of $A$ and $B$ if $d(A, B)$ is the minimum number of edit operations needed to transform $A$ into $B$, where possible operations are:

- insert a character
- delete a character

Example: $A = \text{abc}$ and $B = \text{adef}$

Distance $d(A, B) = 5$: delete $b$, delete $c$, insert $d$, insert $e$, and insert $f$. 
Def. Given two sequences $A[1..n]$ and $B[1..m]$ of letters, $d(A, B)$ is called a edit distance with insert and delete operations of $A$ and $B$ if $d(A, B)$ is the minimum number of edit operations needed to transform $A$ into $B$, where possible operations are:

- insert a character
- delete a character

Example: $A = abc$ and $B = adef$
Def. Given two sequences $A[1..n]$ and $B[1..m]$ of letters, $d(A, B)$ is called a edit distance with insert and delete operations of $A$ and $B$ if $d(A, B)$ is the minimum number of edit operations needed to transform $A$ into $B$, where possible operations are:

- insert a character
- delete a character

Example: $A = abc$ and $B = adef$

Distance $d(A, B) = 5$: delete $b$, delete $c$, insert $d$, insert $e$, and insert $f$. 
Edit distance with three operations (insertions, deletions and replacing)

**Def.** Given two sequences $A[1 .. n]$ and $B[1 .. m]$ of letters, $d(A, B)$ is called a edit distance of $A$ and $B$ if $d(A, B)$ is the minimum number of edit operations needed to transform $A$ into $B$, where possible operations are:

- insert a character
- delete a character
- modify (or replace) a character

Example: $A = abc$ and $B = adef$.

Distance $d(A, B) = 3$: replace $b$ to $d$, replace $c$ to $e$, and insert character $f$. 

Edit distance with three operations (insertions, deletions and replacing)

**Def.** Given two sequences $A[1..n]$ and $B[1..m]$ of letters, $d(A, B)$ is called an edit distance of $A$ and $B$ if $d(A, B)$ is the minimum number of edit operations needed to transform $A$ into $B$, where possible operations are:

- insert a character
- delete a character
- modify (or replace) a character

Example: $A = abc$ and $B = adef$
Edit distance with three operations (insertions, deletions and replacing)

**Def.** Given two sequences $A[1..n]$ and $B[1..m]$ of letters, $d(A, B)$ is called a **edit distance** of $A$ and $B$ if $d(A, B)$ is the minimum number of edit operations needed to transform $A$ into $B$, where possible operations are:

- insert a character
- delete a character
- modify (or replace) a character

**Example:** $A = abc$ and $B = adef$

Distance $d(A, B) = 3$: replace $b$ to $d$, replace $c$ to $e$, and insert character $f$. 
Quiz 5 on Ublearns

- Questions about subsequence, common subsequence, and edit distance
- Deadline: 25 Wed 2023, 11:59PM