Dynamic Programming

DPV Chapter 6, Part 1

Jim Royer

March 6, 2019
Computing binomial coefficients, 1

\[ \text{binom}(n, k) = \begin{cases} 
1, & \text{if } k = 0 \text{ or } k = n; \\
\text{binom}(n - 1, k - 1) + \text{binom}(n - 1, k), & \text{otherwise.}
\end{cases} \]
Computing binomial coefficients, 2

\[ \text{binom}(n, k) = \begin{cases} 
1, & \text{if } k = 0 \text{ or } k = n; \\
\text{binom}(n - 1, k - 1) + \text{binom}(n - 1, k), & \text{otherwise.}
\end{cases} \]
Computing binomial coefficients, 3

Before Memoization

function binom(n, k)
    if k = 0 or k = n then return 1
    else return binom(n − 1, k − 1) + binom(n − 1, k)

After Memoization

function binom(n, k)
    for m ← 0, 1, . . . , n do
        b[m, 0] ← 1; b[m, m] ← 1
    for m ← 2, 3, . . . , n do
        for ℓ ← 1, 2, . . . , m − 1 do
            b[m, ℓ] ← 0
    return helper(n, k)

After Memoization (Continued)

function helper(m, ℓ)
    if b[m, ℓ] = 0 then
        b[m, ℓ] ← helper(m − 1, ℓ − 1) + helper(m − 1, ℓ)
    return b[m, ℓ]

Trace binom(5,3)
Computing binomial coefficients, 4

**Building the Table Directly**

```
function binom(n, k)
    for m ← 0, 1, . . . , n do
        b[m, 0] ← 1;  b[m, m] ← 1
    for m = 2, 3, . . . , n do
        for ℓ = 1, 2, . . . , m − 1 do
            b[m, ℓ] ← b[m − 1, ℓ − 1] + b[m − 1, ℓ]
    return b[n, k]
```

Trace binom(5,3)
Computing binomial coefficients, 5

Going from a recursion to a table-building computation.

Step 1. Give a recursive definition.  
\[(\text{For many problems, this is the hard part.})\]

Step 2. Memoize to exploit repeated subproblems.  
\[(\text{If there are few repeated subproblems, then memoization will not help.})\]

Step 3. Build the table directly to cut down overhead.  
\[(\text{If the answer depends on a small part of the table, then the recursion can be faster.})\]
Making Change—Again, 1

The Making Change Problem (MCP)

**Given:** coin denominations \( d_1 < d_2 < \cdots < d_k \) and an amount \( a \).

**Find:** the smallest collection of coins that is worth amount \( a \).

Example

\( d_1 = 1, d_2 = 4, d_3 = 6; \ a = 8.\)

\( \text{The optimal choice is} \ \{4, 4\}. \)

\( \text{The greedy algorithm produces} \ \{6, 1, 1\}. \)

The Optimal Substructure of MCP

**If:** an optimal solution of the MCP for \( a \) uses a \( d_i \)-coin,

**then:** the rest of the coins give an optimal solution of the MCP for \( a - d_i \).

(Why?)
The Making Change Problem (MCP)

**Given:** coin denominations $d_1 < d_2 < \cdots < d_k$ and an amount $a$.

**Find:** the smallest collection of coins that is worth amount $a$.

$$mcn(a) \equiv \text{the number of coins in an optimal solution to MCP for } a$$

$$mcn(a) = \begin{cases} 
0, \quad & \text{if } a = 0; \\
1 + \min \{ mcn(a - d_i) : d_i \leq a \text{ \& } 1 \leq i \leq k \}, \quad & \text{if } a > 0.
\end{cases}$$

**Example**

- $d_1 = 1$
- $d_2 = 4$
- $d_3 = 6$
- $a = 8$
The Making Change Problem (MCP)

**Given:** coin denominations $d_1 < d_2 < \cdots < d_k$ and an amount $a$.

**Find:** the smallest collection of coins that is worth amount $a$.

$$mcn(a) \equiv \text{the number of coins in an optimal solution to MCP for } a$$

$$mcn(a) = \begin{cases} 
0, & \text{if } a = 0; \\
1 + \min \{ mcn(a - d_i) : d_i \leq a \land 1 \leq i \leq k \}, & \text{if } a > 0.
\end{cases}$$

**Example**

$$mcn(0) = 0$$

$$\begin{align*}
d_1 &= 1 \\
d_2 &= 4 \\
d_3 &= 6 \\
a &= 8
\end{align*}$$
The Making Change Problem (MCP)

**Given:** coin denominations \(d_1 < d_2 < \cdots < d_k\) and an amount \(a\).

**Find:** the smallest collection of coins that is worth amount \(a\).

\[
mcn(a) \equiv \text{the number of coins in an optimal solution to MCP for } a \\
mcn(a) = \begin{cases} 
0, & \text{if } a = 0; \\
1 + \min \left\{ mcn(a - d_i) \mid d_i \leq a \text{ and } 1 \leq i \leq k \right\}, & \text{if } a > 0.
\end{cases}
\]

**Example**

\[
\begin{align*}
mcn(0) &= 0 & = 0 \\
mcn(1) &= 1 + \min \{mcn(0)\} & = 1
\end{align*}
\]

\(d_1 = 1\)

\(d_2 = 4\)

\(d_3 = 6\)

\(a = 8\)
Making Change—Again, 2

The Making Change Problem (MCP)

**Given:** coin denominations \(d_1 < d_2 < \cdots < d_k\) and an amount \(a\).

**Find:** the smallest collection of coins that is worth amount \(a\).

\[
mcn(a) \equiv \text{the number of coins in an optimal solution to MCP for } a
\]

\[
mcn(a) = \begin{cases} 
0, & \text{if } a = 0; \\
1 + \min \{ mcn(a - d_i) : d_i \leq a \text{ and } 1 \leq i \leq k \}, & \text{if } a > 0.
\end{cases}
\]

**Example**

\[
\begin{align*}
mcn(0) &= 0 = 0 \\
mcn(1) &= 1 + \min \{ mcn(0) \} = 1 \\
mcn(2) &= 1 + \min \{ mcn(1) \} = 2 \\
\end{align*}
\]

\(d_1 = 1\)

\(d_2 = 4\)

\(d_3 = 6\)

\(a = 8\)
The Making Change Problem (MCP)

**Given:** coin denominations \( d_1 < d_2 < \cdots < d_k \) and an amount \( a \).

**Find:** the smallest collection of coins that is worth amount \( a \).

\[
mcn(a) \equiv \text{the number of coins in an optimal solution to MCP for } a
\]

\[
mcn(a) = \begin{cases} 
0, & \text{if } a = 0; \\
1 + \min \{ mcn(a - d_i) : d_i \leq a \text{ and } 1 \leq i \leq k \}, & \text{if } a > 0.
\end{cases}
\]

**Example**

\[
\begin{align*}
mcn(0) &= 0 \\
mcn(1) &= 1 + \min \{ mcn(0) \} = 1 \\
mcn(2) &= 1 + \min \{ mcn(1) \} = 2 \\
mcn(3) &= 1 + \min \{ mcn(2) \} = 3 \\
d_1 &= 1 \\
d_2 &= 4 \\
d_3 &= 6 \\
a &= 8
\end{align*}
\]
Making Change—Again, 2

The Making Change Problem (MCP)

**Given:** coin denominations $d_1 < d_2 < \cdots < d_k$ and an amount $a$.

**Find:** the smallest collection of coins that is worth amount $a$.

$$mcn(a) \equiv \text{the number of coins in an optimal solution to MCP for } a$$

$$mcn(a) = \begin{cases} 0, & \text{if } a = 0; \\ 1 + \min \{ mcn(a - d_i) : d_i \leq a \text{ and } 1 \leq i \leq k \}, & \text{if } a > 0. \end{cases}$$

**Example**

$$d_1 = 1$$

$$mcn(0) = 0$$

$$mcn(1) = 1 + \min \{ mcn(0) \} = 1$$

$$d_2 = 4$$

$$mcn(2) = 1 + \min \{ mcn(1) \} = 2$$

$$mcn(3) = 1 + \min \{ mcn(2) \} = 3$$

$$d_3 = 6$$

$$mcn(4) = 1 + \min \{ mcn(3), mcn(0) \} = 1$$

$$a = 8$$

$$mcn(5) = 1 + \min \{ mcn(4), mcn(1) \} = 2$$

$$mcn(6) = 1 + \min \{ mcn(5), mcn(2) \} = 2$$

$$mcn(7) = 1 + \min \{ mcn(6), mcn(3) \} = 3$$

$$mcn(8) = 1 + \min \{ mcn(7), mcn(4) \} = 2$$
Making Change—Again, 2

The Making Change Problem (MCP)

Given: coin denominations $d_1 < d_2 < \cdots < d_k$ and an amount $a$.

Find: the smallest collection of coins that is worth amount $a$.

\[
mcn(a) \equiv \text{the number of coins in an optimal solution to MCP for } a
\]

\[
mcn(a) = \begin{cases} 
0, & \text{if } a = 0; \\
1 + \min \{ mcn(a - d_i) : d_i \leq a \text{ and } 1 \leq i \leq k \}, & \text{if } a > 0.
\end{cases}
\]

Example

\[
\begin{align*}
mcn(0) & = 0 \\
mcn(1) & = 1 + \min \{ mcn(0) \} = 1 \\
mcn(2) & = 1 + \min \{ mcn(1) \} = 2 \\
mcn(3) & = 1 + \min \{ mcn(2) \} = 3 \\
mcn(4) & = 1 + \min \{ mcn(3), mcn(0) \} = 1 \\
mcn(5) & = 1 + \min \{ mcn(4), mcn(1) \} = 2
\end{align*}
\]

\[d_1 = 1\]
\[d_2 = 4\]
\[d_3 = 6\]
\[a = 8\]
Making Change—Again, 2

The Making Change Problem (MCP)

Given: coin denominations \(d_1 < d_2 < \cdots < d_k\) and an amount \(a\).
Find: the smallest collection of coins that is worth amount \(a\).

\[
mcn(a) \equiv \text{the number of coins in an optimal solution to MCP for } a
\]

\[
mcn(a) = \begin{cases} 
0, & \text{if } a = 0; \\
1 + \min \{ mcn(a - d_i) : d_i \leq a \land 1 \leq i \leq k \}, & \text{if } a > 0.
\end{cases}
\]

Example

\[
\begin{array}{c|c|c}
d_1 = 1 & mcn(0) = 0 & \equiv 0 \\
& mcn(1) = 1 + \min \{ mcn(0) \} & = 1 \\
d_2 = 4 & mcn(2) = 1 + \min \{ mcn(1) \} & = 2 \\
& mcn(3) = 1 + \min \{ mcn(2) \} & = 3 \\
d_3 = 6 & mcn(4) = 1 + \min \{ mcn(3), mcn(0) \} & = 1 \\
& mcn(5) = 1 + \min \{ mcn(4), mcn(1) \} & = 2 \\
a = 8 & mcn(6) = 1 + \min \{ mcn(5), mcn(2), mcn(0) \} & = 1
\end{array}
\]
The Making Change Problem (MCP)

**Given:** coin denominations \(d_1 < d_2 < \cdots < d_k\) and an amount \(a\).

**Find:** the smallest collection of coins that is worth amount \(a\).

\[
mcn(a) \equiv \text{the number of coins in an optimal solution to MCP for } a
\]

\[
mcn(a) = \begin{cases} 
0, & \text{if } a = 0; \\
1 + \min \{ mcn(a - d_i) : d_i \leq a \; \& \; 1 \leq i \leq k \}, & \text{if } a > 0.
\end{cases}
\]

**Example**

\[
\begin{align*}
mcn(0) &= 0 & = 0 \\
mcn(1) &= 1 + \min \{ mcn(0) \} & = 1 \\
mcn(2) &= 1 + \min \{ mcn(1) \} & = 2 \\
mcn(3) &= 1 + \min \{ mcn(2) \} & = 3 \\
mcn(4) &= 1 + \min \{ mcn(3), mcn(0) \} & = 1 \\
mcn(5) &= 1 + \min \{ mcn(4), mcn(1) \} & = 2 \\
mcn(6) &= 1 + \min \{ mcn(5), mcn(2), mcn(0) \} & = 1 \\
mcn(7) &= 1 + \min \{ mcn(6), mcn(3), mcn(1) \} & = 2 \\
mcn(8) &= 1 + \min \{ mcn(7), mcn(4), mcn(2), mcn(0) \} & = 2
\end{align*}
\]
The Making Change Problem (MCP)

**Given:** coin denominations \( d_1 < d_2 < \cdots < d_k \) and an amount \( a \).

**Find:** the smallest collection of coins that is worth amount \( a \).

\[
mcn(a) \equiv \text{the number of coins in an optimal solution to MCP for } a
\]

\[
mcn(a) = \begin{cases} 
0, & \text{if } a = 0; \\
1 + \min \{ mcn(a - d_i) : d_i \leq a & 1 \leq i \leq k \}, & \text{if } a > 0.
\end{cases}
\]

**Example**

\[
\begin{align*}
mcn(0) &= 0 \\
mcn(1) &= 1 + \min \{ mcn(0) \} = 1 \\
mcn(2) &= 1 + \min \{ mcn(1) \} = 2 \\
mcn(3) &= 1 + \min \{ mcn(2) \} = 3 \\
mcn(4) &= 1 + \min \{ mcn(3), mcn(0) \} = 1 \\
mcn(5) &= 1 + \min \{ mcn(4), mcn(1) \} = 2 \\
mcn(6) &= 1 + \min \{ mcn(5), mcn(2), mcn(0) \} = 1 \\
mcn(7) &= 1 + \min \{ mcn(6), mcn(3), mcn(1) \} = 2 \\
mcn(8) &= 1 + \min \{ mcn(7), mcn(4), mcn(2) \} = 2
\end{align*}
\]
Making Change—Again, 3

The Making Change Problem (MCP)

Given: coin denominations \(d_1 < d_2 < \cdots < d_k\) and an amount \(a\).
Find: the smallest collection of coins that is worth amount \(a\).

\[
mcn(a) = \begin{cases} 
0, & \text{if } a = 0; \\
1 + \min \{ mcn(a - d_i) : d_i \leq a \text{ and } 1 \leq i \leq k \}, & \text{if } a > 0.
\end{cases}
\]

function \(mcn(d_1, \ldots, d_k; a)\)
integer array \(num[0..a]\)
\(num[0] \leftarrow 0\)
for \(a' \leftarrow 1\) to \(a\) do  // Trace \(mcn(1, 4, 6; 8)\)
    \(num[a'] \leftarrow \infty\)
    for \(i \leftarrow 1\) to \(k\) do
        if \(d_i \leq a'\) then \(num[a'] \leftarrow \min(num[a'], 1 + num[a' - d_i])\)
return \(num[a]\)
Making Change—Again, 4

\[ mcn'(i,a) \equiv \begin{cases} 
\text{the number of coins in an optimal solution to MCP for } a \\
\text{using denominations } d_1, \ldots, d_i \\
0, & \text{if } a = 0; \\
+\infty, & \text{if } i = 0 \text{ and } a > 0; \\
mcn'(i-1,a), & \text{if } 0 < a < d_i; \\
\min\left( mcn'(i-1,a), 1 + mcn'(i,a-d_i) \right), & \text{otherwise} 
\end{cases} \]

**function** \( mcn'(d_1, \ldots, d_k; a) \)

- integer array \( \text{num}[0..k, 0..a] \)
  // Goal: \( \text{num}[i,a'] = mcn(d_1, \ldots, d_i; a') \)
- for \( a' \leftarrow 1 \) to \( a \) do \( \text{num}[0,a'] \leftarrow 0 \)
- for \( i \leftarrow 1 \) to \( k \) do
  // Trace \( mcn'(1,4,6;8) \)
  - \( \text{num}[i,0] \leftarrow 0 \)
  - for \( a' \leftarrow 1 \) to \( k \) do
    - if \( d_i > a' \) then \( \text{num}[i,a'] \leftarrow \text{num}[i-1,a'] \)
    - else \( \text{num}[i,a'] \leftarrow \min( \text{num}[i-1,a'], 1 + \text{num}[i,a'-d_i] ) \)
- return \( \text{num}[k,a] \)
Making Change—Again, 5

Reconstructing the Solution to the MCP

**Given:** $num[i, a'] = \begin{cases} 
\text{the min number of coins of denominations } d_1, \ldots, d_i \text{ need} \\
\text{to make change for amount } a' \text{ where } 0 \leq i \leq k \text{ and } 0 \leq a' \leq a 
\end{cases}$

**Find:** What coins make up the optimal solution.

```python
function reconstruct(d_1, \ldots, d_k; a, num[0..k, 0..a])
    coins ← the empty list
    a' ← a; i ← k
    while a' > 0 do
        if (d_i ≤ a' & num[i, a'] ≠ num[i - 1, a'])
            then Add i to the coins list; a' ← a' − d_i
        else i ← i − 1
    return coins
```
Definition
Suppose $S = a_1, \ldots, a_n$ is a sequence of numbers.

(a) A subsequence of $S$ is a sequence of numbers $a_{i_1}, a_{i_2}, \ldots, a_{i_k}$ such that
    $1 \leq i_1 < i_2 < \cdots < i_k \leq n$.

(b) Such a subsequence is increasing when $a_{i_1} < a_{i_2} < \cdots < a_{i_k}$.

Longest Increasing Subsequence Problem (LISP)

Given: A sequence of numbers.
Find: An increasing subsequence of maximal length.

Example
For $S = 5, 2, 8, 6, 3, 6, 9, 7$; a longest increasing subsequence is: 2, 3, 6, 9.
Longest Increasing Subsequences, 2

Longest Increasing Subsequence Problem (LISP)

**Given:** A sequence of numbers.  

**Find:** A max-length increasing subsequence.

Given $S = a_1, \ldots, a_n$, we can turn this into a graph problem as follows:

Let $V = \{1, \ldots, n\}$, $E = \{(i, j) \mid i < j \& a_i < a_j\}$, and $G = (V, E)$.

$G$ is a dag. \hspace{1cm} (Why?)

\[ \therefore \text{a longest increasing sequence in } S \equiv \text{a longest path in } G. \]

**Example** (for $S = 5, 2, 8, 6, 3, 6, 9, 7$)

![Diagram of dag](Image from DPV)
Longest Increasing Subsequences, 3

Longest Increasing Subsequence Problem (LISP)

**Given:** A sequence of numbers.

**Find:** A max-length increasing subsequence.

\[ G = (V, E), \text{ where } V = \{1, \ldots, n\}, \ E = \{(i, j) : i < j \& a_i < a_j\}. \]

\[
L(j) = \text{the length of a longest increasing subseq. ending at } j \\
= 1 + \max\{ L(i) \mid (i, j) \in E \}. 
\]

**Convention:** \(\max(\emptyset) = 0.\) (So, for example, \(L(1) = 1.\))

**Example (for } S = 5, 2, 8, 6, 3, 6, 9, 7)\]

<table>
<thead>
<tr>
<th>(i)</th>
<th>(1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a_i)</td>
<td>5</td>
</tr>
<tr>
<td>(L(i))</td>
<td>1</td>
</tr>
<tr>
<td>(\text{prev})</td>
<td>0</td>
</tr>
</tbody>
</table>
Longest Increasing Subsequences, 3

Longest Increasing Subsequence Problem (LISP)

Given: A sequence of numbers.

Find: A max-length increasing subsequence.

\( G = (V, E) \), where \( V = \{1, \ldots, n\} \), \( E = \{(i, j) : i < j \& a_i < a_j\} \).

\[
L(j) = \text{the length of a longest increasing subseq. ending at } j \\
= 1 + \max\{ L(i) : (i, j) \in E \}.
\]

Convention: \( \max(\emptyset) = 0 \). (So, for example, \( L(1) = 1 \).)

Example (for \( S = 5, 2, 8, 6, 3, 6, 9, 7 \))

<table>
<thead>
<tr>
<th>( i )</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a_i )</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>( L(i) )</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( \text{prev} )</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

So the length of a LCS is 4 and the LCSs are:

- \( a_2, a_5, a_6, a_8 = 2, 3, 6, 7 \)
- \( a_2, a_5, a_6, a_7 = 2, 3, 6, 9 \)

Note: \( \text{prev}[8] = 6, \text{prev}[6] = 5, \text{prev}[5] = 2, \text{prev}[2] = 0 \)
Longest Increasing Subsequences, 3

Longest Increasing Subsequence Problem (LISP)

Given: A sequence of numbers.
Find: A max-length increasing subsequence.

$G = (V, E)$, where $V = \{1, \ldots, n\}$, $E = \{(i, j) : i < j \& a_i < a_j\}$.

$L(j) = \text{the length of a longest increasing subseq. ending at } j$
$= 1 + \max\{L(i) : (i, j) \in E\}$.

Convention: $\max(\emptyset) = 0$. (So, for example, $L(1) = 1$.)

Example (for $S = 5, 2, 8, 6, 3, 6, 9, 7$)

<table>
<thead>
<tr>
<th>$i$</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_i$</td>
<td>5</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>$L(i)$</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>$\text{prev}$</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

So the length of a LCS is 4 and the LCSs are:

- $a_2, a_5, a_6, a_8 = 2, 3, 6, 7$
  Note: $\text{prev}[8] = 6, \text{prev}[6] = 5, \text{prev}[5] = 2, \text{prev}[2] = 0$
- $a_2, a_5, a_6, a_7 = 2, 3, 6, 9$
Longest Increasing Subsequences, 3

Longest Increasing Subsequence Problem (LISP)

**Given:** A sequence of numbers.

**Find:** A max-length increasing subsequence.

\( G = (V, E), \) where \( V = \{1, \ldots, n\}, E = \{(i, j) : i < j \& a_i < a_j\}. \)

\[
L(j) = \text{the length of a longest increasing subseq. ending at } j
= 1 + \max\{L(i) : (i, j) \in E\}.
\]

**Convention:** \( \max(\emptyset) = 0. \) (So, for example, \( L(1) = 1. \))

**Example** (for \( S = 5, 2, 8, 6, 3, 6, 9, 7 \))

<table>
<thead>
<tr>
<th>( i )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a_i )</td>
<td>5</td>
<td>2</td>
<td>8</td>
<td>6</td>
</tr>
<tr>
<td>( L(i) )</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>prev</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

So the length of a LCS is 4 and the LCSs are:

- \( a_2, a_5, a_6, a_8 = 2, 3, 6, 7 \)
- Note: \( \text{prev}[8] = 6, \text{prev}[6] = 5, \text{prev}[5] = 2, \text{prev}[2] = 0 \)
- \( a_2, a_5, a_6, a_7 = 2, 3, 6, 9 \)
Longest Increasing Subsequences, 3

Longest Increasing Subsequence Problem (LISP)

**Given:** A sequence of numbers.

**Find:** A max-length increasing subsequence.

\[ G = (V, E), \text{ where } V = \{1, \ldots, n\}, \quad E = \{(i, j) : i < j \text{ & } a_i < a_j\}. \]

\[ L(j) = \text{the length of a longest increasing subseq. ending at } j \]
\[ = 1 + \max \{L(i) : (i, j) \in E\}. \]

**Convention:** \( \max(\emptyset) = 0. \) (So, for example, \( L(1) = 1. \))

Example (for \( S = 5, 2, 8, 6, 3, 6, 9, 7 \))

<table>
<thead>
<tr>
<th>( i )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( a_i )</td>
<td>5</td>
<td>2</td>
<td>8</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>( L(i) )</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>( \text{prev} )</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

So the length of a LCS is 4 and the LCSs are:

▶ \( a_2, a_5, a_6, a_8 = 2, 3, 6, 7 \)

Note: \( \text{prev}[8] = 6, \text{prev}[6] = 5, \text{prev}[5] = 2, \text{prev}[2] = 0 \)

▶ \( a_2, a_5, a_6, a_7 = 2, 3, 6, 9 \)
Longest Increasing Subsequences, 3

Longest Increasing Subsequence Problem (LISP)

Given: A sequence of numbers.

Find: A max-length increasing subsequence.

\( G = (V, E) \), where \( V = \{1, \ldots, n\} \), \( E = \{ (i, j) : i < j \& a_i < a_j \} \).

\[ L(j) = \text{the length of a longest increasing subseq. ending at } j = 1 + \max\{ L(i) : (i, j) \in E \}. \]

Convention: \( \max(\emptyset) = 0. \) (So, for example, \( L(1) = 1. \))

Example (for \( S = 5, 2, 8, 6, 3, 6, 9, 7 \))

| \( i \) | 1 | 2 | 3 | 4 | 5 | 6 |
| \( a_i \) | 5 | 2 | 8 | 6 | 3 | 6 |
| \( L(i) \) | 1 | 1 | 2 | 2 | 2 | 3 |
| \( \text{prev} \) | 0 | 0 | 1 | 2 | 2 | 5 |

So the length of a LCS is 4 and the LCSs are:

- \( a_2, a_5, a_6, a_8 = 2, 3, 6, 7 \)
  
  Note: \( \text{prev}[8] = 6, \text{prev}[6] = 5, \text{prev}[5] = 2, \text{prev}[2] = 0 \)

- \( a_2, a_5, a_6, a_7 = 2, 3, 6, 9 \)
Longest Increasing Subsequence Problem (LISP)

**Given:** A sequence of numbers.

**Find:** A max-length increasing subsequence.

\[ G = (V, E), \text{ where } V = \{1, \ldots, n\}, E = \{(i, j) : i < j \land a_i < a_j\}. \]

\[ L(j) = \text{the length of a longest increasing subseq. ending at } j = 1 + \max\{L(i) : (i, j) \in E\}. \]

**Convention:** \( \max(\emptyset) = 0. \) (So, for example, \( L(1) = 1. \))

**Example (for } S = 5, 2, 8, 6, 3, 6, 9, 7)\]

<table>
<thead>
<tr>
<th>(i)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a_i)</td>
<td>5</td>
<td>2</td>
<td>8</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>9</td>
</tr>
<tr>
<td>(L(i))</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>(\text{prev})</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>

So the length of a LCS is 4 and the LCSs are:

- \( a_2, a_5, a_6, a_8 = 2, 3, 6, 7 \)
  - Note: \( \text{prev}[8] = 6, \text{prev}[6] = 5, \text{prev}[5] = 2, \text{prev}[2] = 0 \)
- \( a_2, a_5, a_6, a_7 = 2, 3, 6, 9 \)
Longest Increasing Subsequences, 3

Longest Increasing Subsequence Problem (LISP)

Given: A sequence of numbers.

Find: A max-length increasing subsequence.

$G = (V, E)$, where $V = \{1, \ldots, n\}$, $E = \{(i, j) : i < j \& a_i < a_j\}$.

$L(j) =$ the length of a longest increasing subseq. ending at $j$

$= 1 + \max\{L(i) ; (i, j) \in E\}$.

Convention: $\max(\emptyset) = 0$. (So, for example, $L(1) = 1$.)

Example (for $S = 5, 2, 8, 6, 3, 6, 9, 7$)

<table>
<thead>
<tr>
<th>$i$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a_i$</td>
<td>5</td>
<td>2</td>
<td>8</td>
<td>6</td>
<td>3</td>
<td>6</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>$L(i)$</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>prev</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>

So the length of a LCS is 4 and the LCSs are:

$\rightarrow a_2, a_5, a_6, a_8 = 2, 3, 6, 7$

Note: $\text{prev}[8] = 6, \text{prev}[6] = 5, \text{prev}[5] = 2, \text{prev}[2] = 0$

$\rightarrow a_2, a_5, a_6, a_7 = 2, 3, 6, 9$
Optimal Substructure

A problem has optimal substructure when an optimal solution is made up of optimal solutions to its subproblems.

Examples
(a) Shortest paths in a graph.
(b) Making change.
(c) …

Non-examples
(a) Longest paths in a graph.
(b) 3SAT
(c) …
Longest Common Subsequences, 1

Problem: Longest common subsequence

Given: Strings $s[1..m]$ and $t[1..n]$.
Find: The longest subsequence common to $s$ and $t$.

Example

$s = a \ b \ a \ z \ d \ c$

t = b \ a \ c \ b \ a \ d$
Longest Common Subsequences, 1

Problem: Longest common subsequence

Given: Strings $s[1..m]$ and $t[1..n]$.
Find: The longest subsequence common to $s$ and $t$.

Example

$$s = \text{a} \quad \text{b} \quad \text{a} \quad \text{z} \quad \text{d} \quad \text{c}$$

$$t = \text{b} \quad \text{a} \quad \text{c} \quad \text{b} \quad \text{a} \quad \text{d}$$

Longest Common Subsequences, 2

Simplification

Initially, we’ll just worry about computing the length of the l.c.s.

Subproblems?

\[ LCS[i, j] = \text{the length of the l.c.s. of } s[1..i] \text{ and } t[1..j]. \]

Questions:

1. \( LCS[m, n] = ?? \)
2. \( LCS[0, j] = ?? \)
3. \( LCS[i, 0] = ?? \)
4. \( LCS[i, j] = ?? \) (in terms of \( LCS[i', j'] \) for smaller \( i' \) and \( j' \))
Longest Common Subsequences, 3

$LCS[i, j] =$ the length of the l.c.s. of $s[1..i]$ and $t[1..j]$.

Questions:

1. $LCS[m, n] =$ the answer to the big problem
2 & 3 $LCS[0, j] = LCS[i, 0] = 0$.
4 $LCS[i, j] = ??$ (in terms of $LCS[i', j']$ for smaller $i'$ and $j'$) ($i, j > 0$)

Case: $s[i] = t[j]$. Then?
Case: $s[i] \neq t[j]$. Then?
Longest Common Subsequences, 4

$LCS[i,j] = \text{the length of the l.c.s. of } s[1..i] \text{ and } t[1..j]$.  

$$
= \begin{cases} 
0, & \text{if } i = 0 \text{ or } j = 0 \\
1 + LCS[i-1,j-1], & \text{if } i,j > 0 \text{ and } s[i] = t[j] \\
\max(LCS[i,j-1], LCS[i-1,j]), & \text{otherwise.}
\end{cases}
$$

Exercises for the reader

1. Build the table for $s = \text"abazdc"$ and $t = \text"bacdad"$.
2. Give the algorithm for computing $LCS[0..m,0..n]$.
3. Given the table $LCS[0..m,0..n]$ (and $s$ and $t$), reconstruct the actual longest common subsequence.