Longest Common Subsequence

Kuan-Yu Chen (陳冠宇)

2019/04/23 @ TR-310-1, NTUST

Review

- The problem of matrix-chain multiplication
  - We are given a sequence (chain) $\{A_1, A_2, \cdots, A_n\}$ of n matrices to be multiplied
  - Fully Parenthesized
    - For example, if the chain of matrices is $\{A_1, A_2, A_3, A_4\}$, then we can fully parenthesize the product $A_1A_2A_3A_4$ in five distinct ways
    - $\left(A_1\left(A_2(A_3A_4)\right)\right)$
    - $\left(A_1\left((A_2A_3)A_4\right)\right)$
    - $\left((A_1A_2)(A_3A_4)\right)$
    - $\left(\left(A_1(A_2A_3)\right)A_4\right)$
    - $\left(\left((A_1A_2)A_3\right)A_4\right)$
  - Matrix multiplication is associative, and so all parenthesizations yield the same result

Longest-common-subsequence.

- Biological applications often need to compare the DNA of two (or more) different organisms (有機體)
  - A strand of DNA consists of a string of molecules called ***bases***
    - Adenine, guanine, cytosine, and thymine

- One reason to compare two strands of DNA is to determine how "similar" the two strands are
  - We can define similarity in many different ways
    - S_1 = ACCGGTCGAGTGCGCGGAAGCCGGCCGAA
    - S_2 = GTCGTTCGGAATGCCGTTGCTCTGTAAA

  - A commonly used way to measure the similarity of strands S_1 and S_2 is by finding a third strand S_3 in which the bases in S_3 appear in each of S_1 and S_2
    - The longer the strand S_3 we can find, the more similar S_1 and S_2 are

Longest-common-subsequence..

- Formally, given a sequence $X = \{x_1, x_2, \dots, x_m\}$, another sequence $Z = \{z_1, z_2, \dots, z_k\}$ is a **subsequence** of X if there exists a strictly increasing sequence $\{i_1, i_2, \dots, i_k\}$ of indexes of X such that for all $j = 1,2, \dots, k$, we have $x_{i_j} = z_j$
  - $X = \{A, B, C, B, D, A, B\}$
  - $Z = \{B, C, D, B\}$
  - $\{i_1, i_2, i_3, i_4\} = \{2,3,5,7\}$
    - $x_{i_1} = x_2 = B$
    - $x_{i_2} = x_3 = C$
    - $x_{i_3} = x_5 = D$
    - $x_{i_4} = x_7 = B$

Longest-common-subsequence…

- Given two sequences X and Y, we say that a sequence Z is a ***common subsequence*** of X and Y if Z is a subsequence of both X and Y
  - $X = \{A, B, C, B, D, A, B\}$
  - $Y = \{B, D, C, A, B, A\}$
  - $Z = \{B, C, A\}$ is a common subsequence of both X and Y
  - It is worthy to note that $\{B, C, B, A\}$ is also a common subsequence of both X and Y
  - Since X and Y have no common subsequence of length 5 or greater, thus $\{B, C, B, A\}$ is an **longest-common-subsequence** of both X and Y

DP for LCS.

- For a given sequence $X = \{x_1, x_2, \dots, x_m\}$, we define the i-th prefix of X as $X_i = \{x_1, x_2, \dots, x_i\}$
  - $X = \{A, B, C, B, D, A, B\}$
  - $X_4 = \{A, B, C, B\}$
  - X_0 is the empty sequence

- Let $X = \{x_1, x_2, \dots, x_m\}$ and $Y = \{y_1, y_2, \dots, y_n\}$, and $Z = \{z_1, z_2, \dots, z_k\}$ is an LCS of both X and Y
  - If $x_m = y_n$, then $x_m = y_n = z_k$ and Z_{k-1} is an LCS of X_{m-1} and Y_{n-1}
  - If $x_m \neq y_n$, then $x_m \neq z_k$ implies that Z is an LCS of X_{m-1} and Y
  - If $x_m \neq y_n$, then $y_n \neq z_k$ implies that Z is an LCS of X and Y_{n-1}

DP for LCS..

- Let $X = \{x_1, x_2, \dots, x_m\}$ and $Y = \{y_1, y_2, \dots, y_n\}$, and $Z = \{z_1, z_2, \dots, z_k\}$ is an LCS of both X and Y
  - If $x_m = y_n$, then $x_m = y_n = z_k$ and Z_{k-1} is an LCS of X_{m-1} and Y_{n-1}
  - If $x_m \neq y_n$, then $x_m \neq z_k$ implies that Z is an LCS of X_{m-1} and Y
  - If $x_m \neq y_n$, then $y_n \neq z_k$ implies that Z is an LCS of X and Y_{n-1}

- Let's define $c[i, j]$ to be the length of an LCS of the sequences X_i and Y_j

$$c[i, j] = \begin{cases} 0, if\ i = 0\ or\ j = 0 \\ c[i-1, j-1] + 1, if\ i > 0\ and\ j > 0\ and\ x_i = y_j \\ \max(c[i, j-1], c[i-1, j])\,, if\ i > 0\ and\ j > 0\ and\ x_i \neq y_j \end{cases}$$

DP for LCS…

- Thus, we can construct a LCS-LENGTH procedure
  - Two input sequences $X = \{x_1, x_2, \dots, x_m\}$ and $Y = \{y_1, y_2, \dots, y_n\}$
  - A table $c[0 \dots m, 0 \dots n]$ is to store the values $c[i, j]$
  - A table $b[1 \dots m, 1 \dots n]$ is to help us construct an optimal solution

```
LCS-LENGTH(X, Y)
m = X.length
n = Y.length
let b[1..m, 1..n] and c[0..m, 0..n] be new tables
for i = 1 to m
    c[i, 0] = 0
for j = 0 to n
    c[0, j] = 0
for i = 1 to m
    for j = 1 to n
        if x_i == y_j
            c[i, j] = c[i − 1, j − 1] + 1
            b[i, j] = "↖"
        elseif c[i − 1, j] ≥ c[i, j − 1]
            c[i, j] = c[i − 1, j]
            b[i, j] = "↑"
        else c[i, j] = c[i, j − 1]
            b[i, j] = "←"
return c and b
```

DP for LCS….

$$c[i,j] = \begin{cases} 0, if\ i = 0\ or\ j = 0 \\ c[i-1, j-1] + 1, if\ i > 0\ and\ j > 0\ and\ x_i = y_j \\ \max(c[i, j-1], c[i-1, j])\ , if\ i > 0\ and\ j > 0\ and\ x_i \neq y_j \end{cases}$$

```
LCS-LENGTH(X, Y)
m = X.length
n = Y.length
let b[1..m, 1..n] and c[0..m, 0..n] be new tables
for i = 1 to m
    c[i, 0] = 0
for j = 0 to n
    c[0, j] = 0
for i = 1 to m
    for j = 1 to n
        if x_i == y_j
            c[i, j] = c[i − 1, j − 1] + 1
            b[i, j] = "↖"
        elseif c[i − 1, j] ≥ c[i, j − 1]
            c[i, j] = c[i − 1, j]
            b[i, j] = "↑"
        else c[i, j] = c[i, j − 1]
            b[i, j] = "←"
return c and b
```

Example.

- Given Two input sequences $X = \{A, B, C, B, D, A, B\}$ and $Y = \{B, D, C, A, B, A\}$, please find their LCS

	j	0	1	2	3	4	5	6
i		y_j	B	D	C	A	B	A
0	x_i	0	0	0	0	0	0	0
1	A	0						
2	B	0						
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

```
LCS-LENGTH(X, Y)
m = X.length
n = Y.length
let b[1..m, 1..n] and c[0..m, 0..n] be new tables
for i = 1 to m
    c[i, 0] = 0
for j = 0 to n
    c[0, j] = 0
for i = 1 to m
    for j = 1 to n
        if x_i == y_j
            c[i, j] = c[i-1, j-1] + 1
            b[i, j] = "↖"
        elseif c[i-1, j] ≥ c[i, j-1]
            c[i, j] = c[i-1, j]
            b[i, j] = "↑"
        else c[i, j] = c[i, j-1]
            b[i, j] = "←"
return c and b
```

Example..

- Given Two input sequences $X = \{A, B, C, B, D, A, B\}$ and $Y = \{B, D, C, A, B, A\}$, please find their LCS

	j	0	1	2	3	4	5	6
i		y_j	B	D	C	A	B	A
0	x_i	0	0	0	0	0	0	0
1	A	0	↑ 0					
2	B	0						
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

```
LCS-LENGTH(X, Y)
m = X.length
n = Y.length
let b[1..m, 1..n] and c[0..m, 0..n] be new tables
for i = 1 to m
    c[i, 0] = 0
for j = 0 to n
    c[0, j] = 0
for i = 1 to m
    for j = 1 to n
        if x_i == y_j
            c[i, j] = c[i − 1, j − 1] + 1
            b[i, j] = "↖"
        elseif c[i − 1, j] ≥ c[i, j − 1]
            c[i, j] = c[i − 1, j]
            b[i, j] = "↑"
        else c[i, j] = c[i, j − 1]
            b[i, j] = "←"
return c and b
```

Example...

- Given Two input sequences $X = \{A, B, C, B, D, A, B\}$ and $Y = \{B, D, C, A, B, A\}$, please find their LCS

	j	0	1	2	3	4	5	6
i		y_j	B	D	C	A	B	A
0	x_i	0	0	0	0	0	0	0
1	A	0	↑ 0	↑ 0	↑ 0	↖ 1		
2	B	0						
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

```
LCS-LENGTH(X, Y)
m = X.length
n = Y.length
let b[1..m, 1..n] and c[0..m, 0..n] be new tables
for i = 1 to m
    c[i, 0] = 0
for j = 0 to n
    c[0, j] = 0
for i = 1 to m
    for j = 1 to n
        if x_i == y_j
            c[i, j] = c[i - 1, j - 1] + 1
            b[i, j] = "↖"
        elseif c[i - 1, j] ≥ c[i, j - 1]
            c[i, j] = c[i - 1, j]
            b[i, j] = "↑"
        else c[i, j] = c[i, j - 1]
            b[i, j] = "←"
return c and b
```

Example….

- Given Two input sequences $X = \{A, B, C, B, D, A, B\}$ and $Y = \{B, D, C, A, B, A\}$, please find their LCS

	j	0	1	2	3	4	5	6
i		y_j	B	D	C	A	B	A
0	x_i	0	0	0	0	0	0	0
1	A	0	$\uparrow$ 0	$\uparrow$ 0	$\uparrow$ 0	$\nwarrow$ 1	$\leftarrow$ 1	
2	B	0						
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

LCS-LENGTH(X, Y)

```
m = X.length
n = Y.length
let b[1..m, 1..n] and c[0..m, 0..n] be new tables
for i = 1 to m
    c[i, 0] = 0
for j = 0 to n
    c[0, j] = 0
for i = 1 to m
    for j = 1 to n
        if x_i == y_j
            c[i, j] = c[i − 1, j − 1] + 1
            b[i, j] = "↖"
        elseif c[i − 1, j] ≥ c[i, j − 1]
            c[i, j] = c[i − 1, j]
            b[i, j] = "↑"
        else c[i, j] = c[i, j − 1]
            b[i, j] = "←"
return c and b
```

Example.....

- Given Two input sequences $X = \{A, B, C, B, D, A, B\}$ and $Y = \{B, D, C, A, B, A\}$, please find their LCS

	j	0	1	2	3	4	5	6
i		y_j	B	D	C	A	B	A
0	x_i	0	0	0	0	0	0	0
1	A	0	↑ 0	↑ 0	↑ 0	↖ 1	← 1	↖ 1
2	B	0						
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

```
LCS-LENGTH(X, Y)
m = X.length
n = Y.length
let b[1..m, 1..n] and c[0..m, 0..n] be new tables
for i = 1 to m
    c[i, 0] = 0
for j = 0 to n
    c[0, j] = 0
for i = 1 to m
    for j = 1 to n
        if x_i == y_j
            c[i, j] = c[i − 1, j − 1] + 1
            b[i, j] = "↖"
        elseif c[i − 1, j] ≥ c[i, j − 1]
            c[i, j] = c[i − 1, j]
            b[i, j] = "↑"
        else c[i, j] = c[i, j − 1]
            b[i, j] = "←"
return c and b
```

Example……

- Given Two input sequences $X = \{A, B, C, B, D, A, B\}$ and $Y = \{B, D, C, A, B, A\}$, please find their LCS

	j	0	1	2	3	4	5	6
i		y_j	B	D	C	A	B	A
0	x_i	0	0	0	0	0	0	0
1	A	0	↑ 0	↑ 0	↑ 0	↖ 1	← 1	↖ 1
2	B	0	↖ 1	← 1	← 1	↑ 1	↖ 2	← 2
3	C	0						
4	B	0						
5	D	0						
6	A	0						
7	B	0						

```
LCS-LENGTH(X, Y)
m = X.length
n = Y.length
let b[1..m, 1..n] and c[0..m, 0..n] be new tables
for i = 1 to m
    c[i, 0] = 0
for j = 0 to n
    c[0, j] = 0
for i = 1 to m
    for j = 1 to n
        if x_i == y_j
            c[i, j] = c[i - 1, j - 1] + 1
            b[i, j] = "↖"
        elseif c[i - 1, j] ≥ c[i, j - 1]
            c[i, j] = c[i - 1, j]
            b[i, j] = "↑"
        else c[i, j] = c[i, j - 1]
            b[i, j] = "←"
return c and b
```

Example……

- Given Two input sequences $X = \{A, B, C, B, D, A, B\}$ and $Y = \{B, D, C, A, B, A\}$, please find their LCS

	j	0	1	2	3	4	5	6
i		y_j	B	D	C	A	B	A
0	x_i	0	0	0	0	0	0	0
1	A	0	↑ 0	↑ 0	↑ 0	↖ 1	← 1	↖ 1
2	B	0	↖ 1	← 1	← 1	↑ 1	↖ 2	← 2
3	C	0	↑ 1	↑ 1	↖ 2	← 2	↑ 2	↑ 2
4	B	0	↖ 1	↑ 1	↑ 2	↑ 2	↖ 3	← 3
5	D	0	↑ 1	↖ 2	↑ 2	↑ 2	↑ 3	↑ 3
6	A	0	↑ 1	↑ 2	↑ 2	↖ 3	↑ 3	↖ 4
7	B	0	↖ 1	↑ 2	↑ 2	↑ 3	↖ 4	↑ 4

```
LCS-LENGTH(X, Y)
m = X.length
n = Y.length
let b[1..m, 1..n] and c[0..m, 0..n] be new tables
for i = 1 to m
    c[i, 0] = 0
for j = 0 to n
    c[0, j] = 0
for i = 1 to m
    for j = 1 to n
        if x_i == y_j
            c[i, j] = c[i − 1, j − 1] + 1
            b[i, j] = "↖"
        elseif c[i − 1, j] ≥ c[i, j − 1]
            c[i, j] = c[i − 1, j]
            b[i, j] = "↑"
        else c[i, j] = c[i, j − 1]
            b[i, j] = "←"
return c and b
```

DP for LCS.....

- The PRINT-LCS is used to print the path
  - The initial call is PRINT-LCS$(b, X, X.length, Y.length)$

i	j	0	1	2	3	4	5	6
		y_j	B	D	C	A	B	A
0	x_i	0	0	0	0	0	0	0
1	A	0	↑ 0	↑ 0	↑ 0	↖ 1	← 1	↖ 1
2	B	0	↖ 1	← 1	← 1	↑ 1	↖ 2	← 2
3	C	0	↑ 1	↑ 1	↖ 2	← 2	↑ 2	↑ 2
4	B	0	↖ 1	↑ 1	↑ 2	↑ 2	↖ 3	← 3
5	D	0	↑ 1	↖ 2	↑ 2	↑ 2	↑ 3	↑ 3
6	A	0	↑ 1	↑ 2	↑ 2	↖ 3	↑ 3	↖ 4
7	B	0	↖ 1	↑ 2	↑ 2	↑ 3	↖ 4	↑ 4

```
PRINT-LCS(b, X, i, j)
if i == 0 or j == 0
    return
if b[i, j] == "↖"
    PRINT-LCS(b, X, i - 1, j - 1)
    print x_i
elseif b[i, j] == "↑"
    PRINT-LCS(b, X, i - 1, j)
else PRINT-LCS(b, X, i, j - 1)
```

Questions?

kychen@mail.ntust.edu.tw