Dynamic Programming

2017년 10월 11일 수요일

오후 1:40

· Four-step of developing DP algo

① characterize the value of an optimal solution

② Recursively define the value of an optimal solution (bottom-up fashion)

③ Compute the value of an optimal solution

④ Construct an optimal solution from computed information

Problem 1. Longest common Subsequence (LCS)

ex) ENTROPY
THORNY

* subseq
TOY
TRY

* sub ~~seq~~ string
One letter

Problem

Given two seq $A = \langle a_1, \ldots, a_m \rangle$ and

$B = \langle b_1, \ldots b_n \rangle$, find a max-length common subseq. of A & B.

subproblem

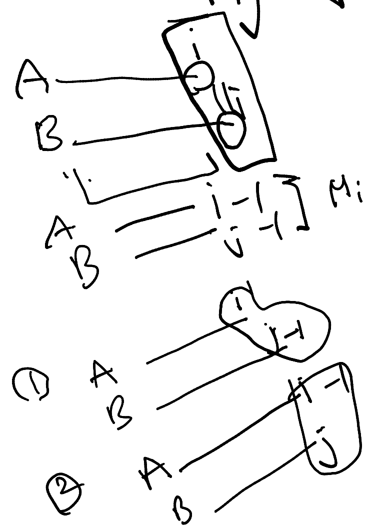
$a_1 \ \ldots \ a_i$] prefix of string
$b_1 \ \ldots \ b_j$

① let M_{ij} ← length of a LCS of the subproblem. $\left[\begin{matrix} a_1 \sim a_i \\ b_1 \sim b_j \end{matrix}\right.$

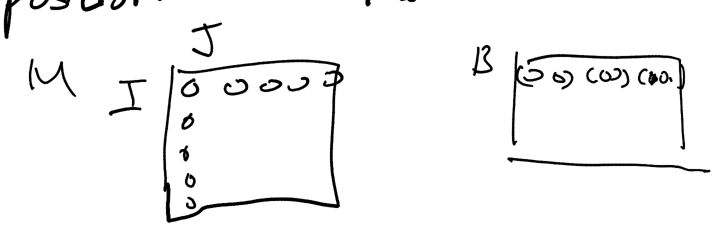
* at end we what $\boxed{M_{nm}}$

②

$$M_{ij} \leftarrow \begin{cases} 0 & \text{if } i=0 \text{ or } j=0 \\ M_{i-1,j-1}+1 & \text{if } i>0,\ j>0\ \ a_i=b_j \ \text{(match)} \\ \text{Max}\{M_{i-1,j},\ M_{i,j-1}\} & \text{if } i>0,\ j>0\ \ a_i \neq b_j \end{cases}$$



B_{ij} ← postion of "last match"



Alg LCS

<u>for</u> $j \leftarrow 0$ to n <u>do</u>

$M_{0,j} \leftarrow 0$ $\quad B_{0,j} \leftarrow (0,0)$

<u>endfor</u>

<u>for</u> $i \leftarrow 1$ to m <u>do</u>

$M_{i,0} \leftarrow 0$ and $B_{i,0} \leftarrow (0,0)$

<u>for</u> $j \leftarrow 1$ to n do

<u>if</u> $a_i = b_j$ /* match */

<u>then</u> $\begin{cases} M_{ij} \leftarrow M_{i-1,j-1} + 1 \\ B_{ij} \leftarrow (i,j) \end{cases}$

~~else~~ <u>if</u> $M_{i-1,j} > M_{i,j-1}$

<u>then</u> $\begin{cases} M_{ij} \leftarrow M_{i-1,j} \\ B_{ij} \leftarrow B_{i-1,j} \end{cases}$

<u>elseif</u> $M_{i,j-1} > M_{i-1,j}$

<u>Then</u> $\begin{cases} M_{ij} \leftarrow M_{i,j-1} \\ B_{ij} \leftarrow B_{i,j-1} \end{cases}$

<u>endif</u>

<u>endfor</u>

<u>endfor</u>

/* Read back Pointer */

$j \leftarrow n$, $i \leftarrow m$

<u>While</u> $M_{ij} > 0$ <u>do</u>

print B_{ij}

$(i,j) \leftarrow B_{ij} - (1,1)$

<u>endwhile</u>