# Approximate String Matching

* String defn.

$$\bar{x} = x_0 x_1 \cdots x_{m-1}, \quad \bar{y} = y_0 y_1 \cdots y_{n-1}$$

* Edit costs

| | | |
|---|---|---|
| match | $x_i = y_j$ | $d(x_i, y_j)$ |
| substitution | $x_i \neq y_j$ | $d(x_i, y_j)$ |
| deletion | $x_i \sim \varepsilon$ | $d(x_i, \varepsilon)$ |
| insertion | $\varepsilon \sim y_j$ | $d(\varepsilon, y_j)$ |

Symbols $x_i, y_j \in$ alphabet, Symbol $\varepsilon$ unobservable (bookkeeping only)

* String alignment, cost

$$x_0 - x_1 x_2 \cdots x_{m-1}$$
$$y_0 y_1 - y_2 \cdots y_{n-1}$$

match/sub | insertion | deletion | match/sub

$$d(\bar{x}, \bar{y}) = \sum_{\text{edits}} d(x_i, y_j)$$

* Optimal string alignment

$$d(\bar{x}, \bar{y}) = \min_{\text{all possible edit segs.}} \sum d(x_i, y_j)$$

# Solution

$$d(\varepsilon x_0 x_1 \cdots x_{m-1}, \varepsilon y_0 y_1 \cdots y_{n-1}) = \min \{$$

DEL $\quad d(\varepsilon x_0 x_1 \cdots x_{m-2}, \varepsilon y_0 y_1 \cdots y_{n-1}) + d(x_{m-1}, \varepsilon)$ ,

MATCH/SUB $\quad d(\varepsilon x_0 x_1 \cdots x_{m-2}, \varepsilon y_0 y_1 \cdots y_{n-2}) + d(x_{m-1}, y_{n-1})$ ,

INS $\quad d(\varepsilon x_0 x_1 \cdots x_{m-1}, \varepsilon y_0 y_1 \cdots y_{n-2}) + d(\varepsilon, y_{n-1})$ }

1. Apply recursively ⇒ lots of overlapped computation
2. Use memoization to store results (matrix cache)
3. Switch to iteration (forward computation)
4. Reduce mem usage if necessary (vector cache)

Cost matrix: $D[m+1][n+1]$

$D(0,0) = 0$ $\qquad D(i,0) = D(i-1,0) + d(x_i, \varepsilon)$ DEL

$\qquad\qquad\quad D(0,j) = D(0,j-1) + d(\varepsilon, y_j)$ INS

$$D(i,j) = \min \begin{cases} D(i-1,j) + d(x_i, \varepsilon) & \text{DEL} \\ D(i-1,j-1) + d(x_i, y_j) & \text{MATCH/SUB} \\ D(i,j-1) + d(\varepsilon, y_j) & \text{INS} \end{cases}$$

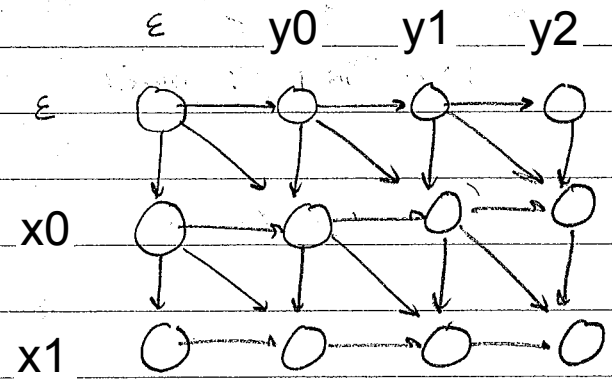
link matrix: $L[m+1][n+1]$

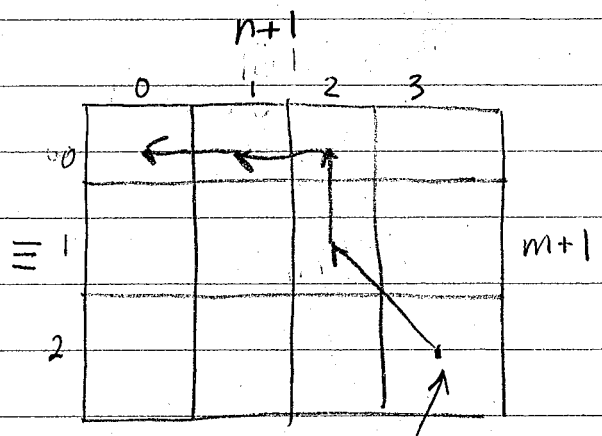
$L(i,j) =$ MATCH/SUB, DEL aka UP INS

DIAG VERT HORZ

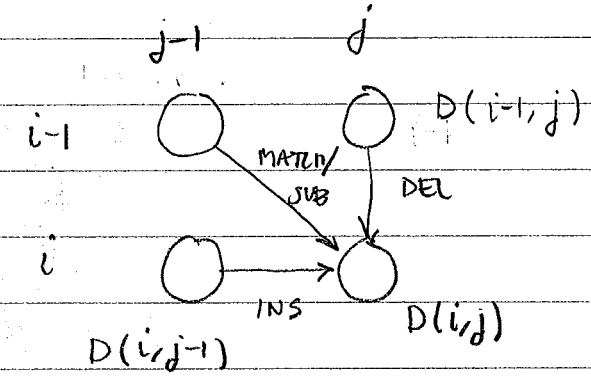
shortest path thru edit graph

## ✱ Graph view of computation



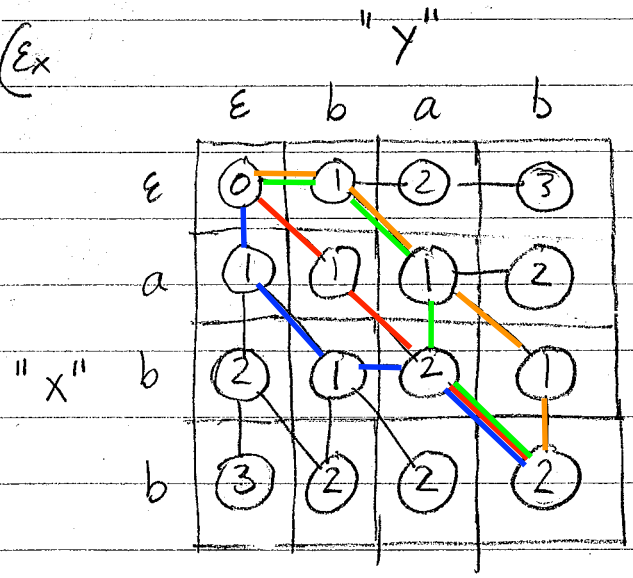

$D(m,n) = d(\bar{x}, \bar{y})$




record optimal move(s) in link matrix

## Levenshtein edit cost

(Ex




$$d(x_i, y_j) = \begin{cases} 0 & \text{MATCH} \\ 1 & \text{SUB} \end{cases}$$

$d(x_i, \epsilon) = 1$ DEL

$d(\epsilon, y_j) = 1$ INS

(counts num. edits)

choose in order of DIAG, VERT, HORZ

① a b b / b a b

② – a b b / b a – b

③ a b – b / – b a b

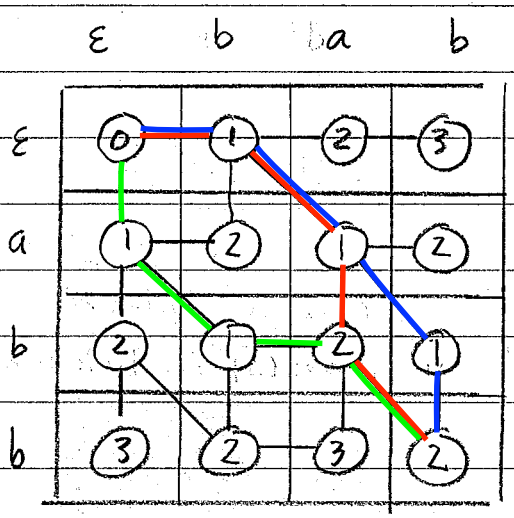
④ – a b b / b a b –

obtain alignments using backtracking and stack.

## ☆ Longest Common Subsequence (LCS)

$$d(x_i, y_j) = \begin{cases} 0 & \text{MATCH} \\ \infty & \text{SUB} \end{cases} \Rightarrow \text{(SUB not allowed)} \quad \text{MATCH}$$

$d(x_i, \varepsilon) = 1$ DEL $\qquad d(\varepsilon, y_j) = 1$ INS




② a b − b
− b a b

① − a b b
b a − b

③ − a b b
b a b −

choose in order of DIAG, VERT, HORZ

NOTE! ⇓

$$D(m,n) = m + n - 2|LCS|$$

$$|LCS| = (m+n - D(m,n))/2$$

(Ex 2 = (3+3−2)/2

conv. to the two matched symbols

## ☆ Lab 8 : Diff