14.2.3
Edit distance: The algorithm

Edit distance

Basic observation

Let $\boldsymbol{X} = \boldsymbol{\alpha x}$ and $\boldsymbol{Y} = \boldsymbol{\beta y}$

$\boldsymbol{\alpha}, \boldsymbol{\beta}$: strings.

$\boldsymbol{x}$ and $\boldsymbol{y}$ single characters.

Think about optimal edit distance between $\boldsymbol{X}$ and $\boldsymbol{Y}$ as alignment, and consider last column of alignment of the two strings:

$\boldsymbol{\alpha}$	$\boldsymbol{x}$
$\boldsymbol{\beta}$	$\boldsymbol{y}$

or

$\boldsymbol{\alpha}$	$\boldsymbol{x}$
$\boldsymbol{\beta y}$	

or

$\boldsymbol{\alpha x}$	
$\boldsymbol{\beta}$	$\boldsymbol{y}$

Observation 14.5.

Prefixes must have optimal alignment!

Problem Structure

Observation 14.6.

Let $X = x_1x_2 \cdots x_m$ and $Y = y_1y_2 \cdots y_n$. If (m, n) are not matched then either the mth position of X remains unmatched or the nth position of Y remains unmatched.

1. Case x_m and y_n are matched.
   1. Pay mismatch cost $\alpha_{x_m y_n}$ plus cost of aligning strings $x_1 \cdots x_{m-1}$ and $y_1 \cdots y_{n-1}$
2. Case x_m is unmatched.
   1. Pay gap penalty plus cost of aligning $x_1 \cdots x_{m-1}$ and $y_1 \cdots y_n$
3. Case y_n is unmatched.
   1. Pay gap penalty plus cost of aligning $x_1 \cdots x_m$ and $y_1 \cdots y_{n-1}$

Subproblems and Recurrence

$x_1 \dots x_{i-1}$	x_i
$y_1 \dots y_{j-1}$	y_j

or

$x_1 \dots x_{i-1}$	x
$y_1 \dots y_{j-1} y_j$	

or

$x_1 \dots x_{i-1} x_i$	
$y_1 \dots y_{j-1}$	y_j

Optimal Costs

Let $\mathbf{Opt}(i, j)$ be optimal cost of aligning $x_1 \cdots x_i$ and $y_1 \cdots y_j$. Then

$$\mathbf{Opt}(i,j) = \min \begin{cases} \alpha_{x_i y_j} + \mathbf{Opt}(i-1, j-1), \\ \delta + \mathbf{Opt}(i-1, j), \\ \delta + \mathbf{Opt}(i, j-1) \end{cases}$$

Base Cases: $\mathbf{Opt}(i, 0) = \delta \cdot i$ and $\mathbf{Opt}(0, j) = \delta \cdot j$

Subproblems and Recurrence

$x_1 \dots x_{i-1}$	x_i
$y_1 \dots y_{j-1}$	y_j

or

$x_1 \dots x_{i-1}$	x
$y_1 \dots y_{j-1} y_j$	

or

$x_1 \dots x_{i-1} x_i$	
$y_1 \dots y_{j-1}$	y_j

Optimal Costs

Let $\mathbf{Opt}(i, j)$ be optimal cost of aligning $x_1 \cdots x_i$ and $y_1 \cdots y_j$. Then

$$\mathbf{Opt}(i,j) = \min \begin{cases} \alpha_{x_i y_j} + \mathbf{Opt}(i-1, j-1), \\ \delta + \mathbf{Opt}(i-1, j), \\ \delta + \mathbf{Opt}(i, j-1) \end{cases}$$

Base Cases: $\mathbf{Opt}(i, 0) = \delta \cdot i$ and $\mathbf{Opt}(0, j) = \delta \cdot j$

Recursive Algorithm

Assume X is stored in array $A[1..m]$ and Y is stored in $B[1..n]$
Array $COST$ stores cost of matching two chars. Thus $COST[a, b]$ give the cost of matching character a to character b.

$EDIST(A[1..m], B[1..n])$
    If $(m = 0)$ return $n\delta$
    If $(n = 0)$ return $m\delta$
    $m_1 = \delta + EDIST(A[1..(m-1)], B[1..n])$
    $m_2 = \delta + EDIST(A[1..m], B[1..(n-1)]))$
    $m_3 = COST[A[m], B[n]] + EDIST(A[1..(m-1)], B[1..(n-1)])$
    return $\min(m_1, m_2, m_3)$

Example: DEED and DREAD

	ϵ	***D***	***R***	***E***	***A***	***D***
ϵ						
D						
E						
E						
D						

Example: DEED and DREAD

	ε	***D***	***R***	***E***	***A***	***D***
ε	0	1	2	3	4	5
D	1					
E	2					
E	3					
D	3					

Example: DEED and DREAD

	ε	***D***	***R***	***E***	***A***	***D***
ε	0	1	2	3	4	5
D	1	0	1	2	3	4
E	2					
E	3					
D	3					

Example: DEED and DREAD

	ε	***D***	***R***	***E***	***A***	***D***
ε	0	1	2	3	4	5
D	1	0	1	2	3	4
E	2	1	1	1	2	3
E	3					
D	3					

Example: DEED and DREAD

	ε	***D***	***R***	***E***	***A***	***D***
ε	0	1	2	3	4	5
D	1	0	1	2	3	4
E	2	1	1	1	2	3
E	3	2	2	1	2	3
D	3					

Example: DEED and DREAD

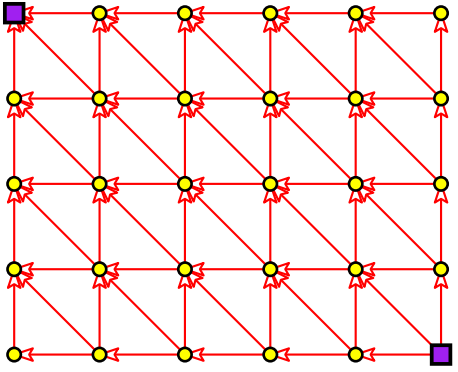
	ε	***D***	***R***	***E***	***A***	***D***
ε	0	1	2	3	4	5
D	1	0	1	2	3	4
E	2	1	1	1	2	3
E	3	2	2	1	2	3
D	3	3	3	2	2	2

D	R	E	A	D
D	E	E		D

Example: DEED and DREAD

	ϵ	***D***	***R***	***E***	***A***	***D***
ϵ	0	1	2	3	4	5
D	1	0	1	2	3	4
E	2	1	1	1	2	3
E	3	2	2	1	2	3
D	3	3	3	2	2	2

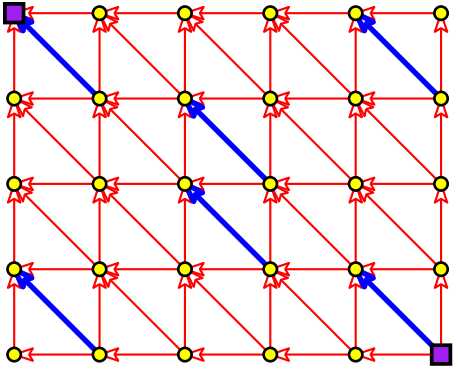
D	R	E	A	D
D	E	E		D



Example: DEED and DREAD

	ε	***D***	***R***	***E***	***A***	***D***
ε	0	1	2	3	4	5
D	1	0	1	2	3	4
E	2	1	1	1	2	3
E	3	2	2	1	2	3
D	3	3	3	2	2	2

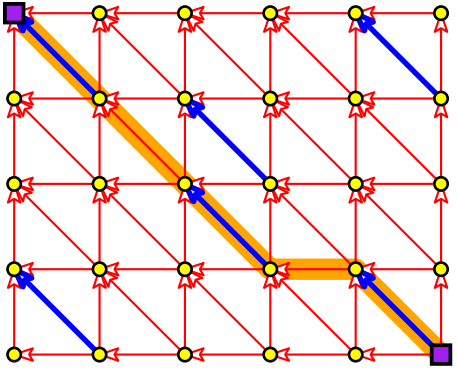
D	R	E	A	D
D	E	E		D



Example: DEED and DREAD

	ε	***D***	***R***	***E***	***A***	***D***
ε	0	1	2	3	4	5
D	1	0	1	2	3	4
E	2	1	1	1	2	3
E	3	2	2	1	2	3
D	3	3	3	2	2	2

D	R	E	A	D
D	E	E		D



THE END

...

(for now)