



Problem: How similar are two strings?

- Spell correction

- The user typed an unknown word "graffe"

We want to know what the user meant?

- graf
- graft
- grail
- giraffe

- Computational Biology

- Align two sequences of nucleotides

```

AGGCTATCACCTGACCTCCAGGCCGATGCC
TAGCTATCACGACCGCGGTCGATTTGCCGAC
  
```

- Resulting alignment:

```

-AGGCTATCACCTGACCTCCAGGCCGA--TGCCC---
TAG-CTATCAC--GACCGC--GGTCGATTTGCCGAC
  
```



Other uses of Edit Distance in NLP

- Entity Coreference
 - **Stanford President John Hennessy** announced yesterday
 - for **Stanford University President John Hennessy**



Min Edit Distance= MED

- The **minimum edit distance** between two strings is the minimum number of editing operations needed to transform one into the other
 - Insertion
 - Deletion
 - Substitution



Minimum Edit Distance

- **alignment:** A correspondence (مطابقت) between characters

I	N	T	E	*	N	T	I	O	N
*	E	X	E	C	U	T	I	O	N



Minimum Edit Distance

I N T E * N T I O N
| | | | | | | | |
* E X E C U T I O N
d s s i s

- If each operation has cost of 1
 - Distance between these is 5
- If substitutions cost 2 (Levenshtein distance)
 - Distance between them is 8



i n t e n t i o n	← delete i
n t e n t i o n	← substitute n by e
e t e n t i o n	← substitute t by x
e x e n t i o n	← insert u
e x e n u t i o n	← substitute n by c
e x e c u t i o n	

Figure 2.16 Path from *intention* to *execution*.



Alignment in Computational Biology

- Given a sequence of bases

AGGCTATCACCTGACCTCCAGGCCGATGCC
TAGCTATCACGACCGCGGTCGATTTGCCCGAC

- An alignment:

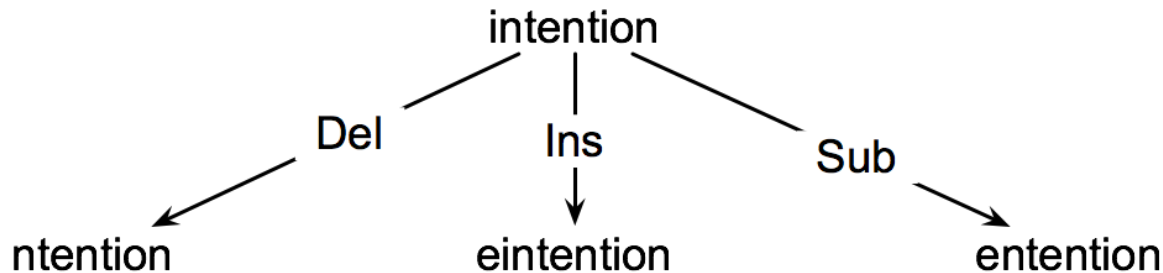
-AGGCTATCACCTGACCTCCAGGCCGA--TGCCC---
TAG-CTATCAC--GACCGC--GGTCGATTTGCCCGAC

- Given two sequences, align each letter to a letter or gap



How to find the Min Edit Distance?

- Searching for a path (sequence of edits) from the start string to the final string:
 - **Initial state:** the word we're transforming
 - **Operators:** insert, delete, substitute
 - **Goal state:** the word we're trying to get to
 - **Path cost:** what we want to minimize: the number of edits





Minimum Edit as Search

- But the space of all edit sequences is huge! (exhaustive search)
 - We can't afford to navigate naively
 - Lots of distinct paths appear at the same state.
 - We don't have to keep track of all of them



Defining Min Edit Distance

- For two strings
 - X of length n
 - Y of length m
- We define $D(i,j)$
 - the edit distance between $X[1..i]$ and $Y[1..j]$
 - i.e., the first i characters of X and the first j characters of Y
 - The edit distance between X and Y is thus $D(n,m)$



Dynamic Programming for Minimum Edit Distance

- **Dynamic programming:** A tabular computation of $D(n,m)$
- Solving problems by combining solutions to subproblems.
- Bottom-up
 - We compute $D(i,j)$ for small i,j
 - And compute larger $D(i,j)$ based on previously computed smaller values
 - i.e., compute $D(i,j)$ for all i ($0 < i < n$) and j ($0 < j < m$)



Defining Min Edit Distance (Levenshtein)

- Initialization

$$D(i, 0) = i$$

$$D(0, j) = j$$

$$D[i, j] = \min \begin{cases} D[i-1, j] + \text{del-cost}(\text{source}[i]) \\ D[i, j-1] + \text{ins-cost}(\text{target}[j]) \\ D[i-1, j-1] + \text{sub-cost}(\text{source}[i], \text{target}[j]) \end{cases}$$

- Recurrence Relation:

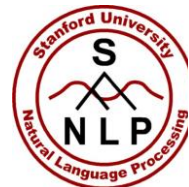
For each $i = 1 \dots M$

For each $j = 1 \dots N$

$$D(i, j) = \min \begin{cases} D(i-1, j) + 1 \\ D(i, j-1) + 1 \\ D(i-1, j-1) + \begin{cases} 2; & \text{if } X(i) \neq Y(j) \\ 0; & \text{if } X(i) = Y(j) \end{cases} \end{cases}$$

- Termination:

$D(N, M)$ is distance



function MIN-EDIT-DISTANCE(*source*, *target*) **returns** *min-distance*

$n \leftarrow \text{LENGTH}(\textit{source})$

$m \leftarrow \text{LENGTH}(\textit{target})$

Create a distance matrix $D[n+1, m+1]$

Initialization: the zeroth row and column is the distance from the empty string

$D[0,0] = 0$

for each row i **from** 1 **to** n **do**

$D[i,0] \leftarrow D[i-1,0] + \textit{del-cost}(\textit{source}[i])$

for each column j **from** 1 **to** m **do**

$D[0,j] \leftarrow D[0,j-1] + \textit{ins-cost}(\textit{target}[j])$

Recurrence relation:

for each row i **from** 1 **to** n **do**

for each column j **from** 1 **to** m **do**

$D[i,j] \leftarrow \text{MIN}(D[i-1,j] + \textit{del-cost}(\textit{source}[i]),$
 $D[i-1,j-1] + \textit{sub-cost}(\textit{source}[i], \textit{target}[j]),$
 $D[i,j-1] + \textit{ins-cost}(\textit{target}[j]))$

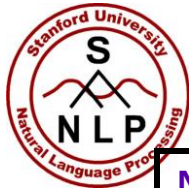
Termination

return $D[n,m]$



The Edit Distance Table

N	9									
O	8									
I	7									
T	6									
N	5									
E	4									
T	3									
N	2									
I	1									
#	0	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N



The Edit Distance Table

N	9										
O	8										
I	7										
T	6										
N	5										
E	4										
T	3										
N	2										
I	1										
#	0	1	2	3	4	5	6	7	8	9	
	#	E	X	E	C	U	T	I	O	N	

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases}$$

 A red arrow points from the top-left towards the cell containing 'I' in the first column and '1' in the second column of the table.



Edit Distance

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases}$$

N	9									
O	8									
I	7									
T	6									
N	5									
E	4									
T	3									
N	2									
I	1									
#	0	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N



The Edit Distance Table

N	9	8	9	10	11	12	11	10	9	8
O	8	7	8	9	10	11	10	9	8	9
I	7	6	7	8	9	10	9	8	9	10
T	6	5	6	7	8	9	8	9	10	11
N	5	4	5	6	7	8	9	10	11	10
E	4	3	4	5	6	7	8	9	10	9
T	3	4	5	6	7	8	7	8	9	8
N	2	3	4	5	6	7	8	7	8	7
I	1	2	3	4	5	6	7	6	7	8
#	0	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N