

# CSE 417: Algorithms and Computational Complexity

## 7,8: Dyn. Programming, IV String Edit Distance

Winter 2005  
W. L. Ruzzo

1

## Sequence Comparison: Edit Distance

- Given:
  - Two strings  $A=a_1 a_2 \dots a_n$  and  $B=b_1 b_2 \dots b_m$
- Find: The minimum number of edit steps to transform  $A$  into  $B$  where a step can be:
  - insert a single character
  - delete a single character
  - substitute one character by another
  - (you can copy a single character for free)

2

### Example

- A = castle
- B = chattel

A		B
c	→	c
	insert	h
a	→	a
s	subst	t
t	→	t
l	delete	
e	→	e
	insert	l

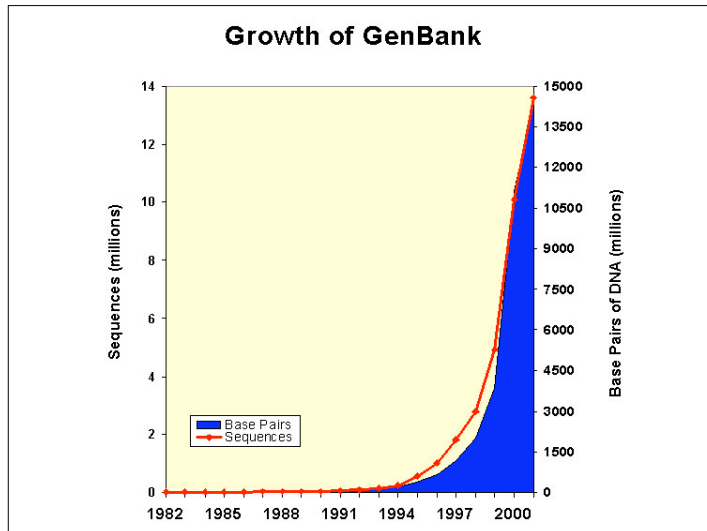
Cost: 4

3

### Applications

- "diff" utility – where do two files differ
- Version control & patch distribution – save/send only changes
- Molecular biology
  - Similar sequences often have similar origin and function
  - Similarity often recognizable despite millions or billions of years of evolutionary divergence

4



## Recursive Solution

- **Sub-problems:** Edit distance problems for all prefixes of **A** and **B** that don't include all of both **A** and **B**
- Let  $D(i,j)$  be the number of edits required to transform  $a_1 a_2 \dots a_i$  into  $b_1 b_2 \dots b_j$
- Clearly  $D(0,0)=0$

6

## Computing $D(n,m)$

- Imagine how best sequence handles the last characters  $a_n$  and  $b_m$
- If best sequence of operations
  - deletes  $a_n$  then  $D(n,m)=D(n-1,m)+1$
  - inserts  $b_m$  then  $D(n,m)=D(n,m-1)+1$
  - replaces  $a_n$  by  $b_m$  then  $D(n,m)=D(n-1,m-1)+1$
  - matches  $a_n$  and  $b_m$  then  $D(n,m)=D(n-1,m-1)$

7

## Recursive algorithm $D(n,m)$

```

if n=0 then
  return (m)
elseif m=0 then
  return(n)
else
  if  $a_n=b_m$  then
    replace-cost=0
  else
    replace-cost=1
  endif
  return(min{  $D(n-1, m) + 1,$ 
              $D(n, m-1) + 1,$ 
              $D(n-1, m-1) + \text{replace-cost}$ })

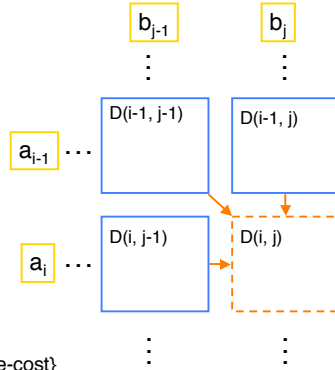
```

8

## Dynamic Programming

```

for j = 0 to m; D(0,j) ← j; endfor
for i = 1 to n; D(i,0) ← i; endfor
for i = 1 to n
  for j = 1 to m
    if ai=bj then
      replace-cost ← 0
    else
      replace-cost ← 1
    endif
    D(i,j) ← min { D(i-1, j) + 1,
                  D(i, j-1) + 1,
                  D(i-1, j-1) + replace-cost }
  endfor
endfor
  
```



9

## Example run with AGACATTG and GAGTTA

	0	A	G	A	C	A	T	T	G
0	0								
G 1									
A 2									
G 3									
T 4									
T 5									
A 6									

10

## Example run with AGACATTG and GAGTTA

	0	A	G	A	C	A	T	T	G
0	0	1	2	3	4	5	6	7	8
G 1	1								
A 2	2								
G 3	3								
T 4	4								
T 5	5								
A 6	6								

11

## Example run with AGACATTG and GAGTTA

	0	A	G	A	C	A	T	T	G
0	0	1	2	3	4	5	6	7	8
G 1	1	1	1	2	3	4	5	6	7
A 2	2								
G 3	3								
T 4	4								
T 5	5								
A 6	6								

12

### Example run with AGACATTG and GAGTTA

	0	A	G	A	C	A	T	T	G
0	0	1	2	3	4	5	6	7	8
G 1	1	1	1	2	3	4	5	6	7
A 2	2	1	2	1					
G 3	3								
T 4	4								
T 5	5								
A 6	6								

13

### Example run with AGACATTG and GAGTTA

	0	A	G	A	C	A	T	T	G
0	0	1	2	3	4	5	6	7	8
G 1	1	1	1	2	3	4	5	6	7
A 2	2	1	2	1	2	3	4	5	6
G 3	3	2	1	2	2	3	4	5	5
T 4	4								
T 5	5								
A 6	6								

14

### Example run with AGACATTG and GAGTTA

	0	A	G	A	C	A	T	T	G
0	0	1	2	3	4	5	6	7	8
G 1	1	1	1	2	3	4	5	6	7
A 2	2	1	2	1	2	3	4	5	6
G 3	3	2	1	2	2	3	4	5	5
T 4	4	3	2	2	3	3	3	4	5
T 5	5	4	3	3	3	4	3	3	4
A 6	6	5	4	3	4	3	4	4	4

15

### Example run with AGACATTG and GAGTTA

		A	G	A	C	A	T	T	G
0	0	1	2	3	4	5	6	7	8
1	1	1	1	2	3	4	5	6	7
2	2	1	2	1	2	3	4	5	6
3	3	2	1	2	2	3	4	5	5
4	4	3	2	2	3	3	3	4	5
5	5	4	3	3	3	4	3	3	4
6	6	5	4	3	4	3	4	4	4

16

## Example run with AGACATTG and GAGTTA

	A	G	A	C	A	T	T	G	
V	0	1	2	3	4	5	6	7	8
L	1	1	1	2	3	4	5	6	7
I	2	1	2	1	2	3	4	5	6
G	3	2	1	2	2	3	4	5	5
T	4	3	2	2	3	3	3	4	5
A	5	4	3	3	3	4	3	3	4
G	6	5	4	3	4	3	4	4	4

17

## Reading off the operations

- Follow the sequence and use color/  
direction of arrows to tell what operation  
was performed.

- ← Insert
- ↑ Delete
- ↖ Copy or substitute

18