Variations on edit distance

Ben Langmead

JOHNS HOPKINS
WHITING SCHOOL
of ENGINEERING

Department of Computer Science

You are free to use these slides. If you do, please sign the guestbook (www.langmead-lab.org/teaching-materials), or email me (ben.langmead@gmail.com) and tell me briefly how you're using them. For original Keynote files, email me.

Edit distance: dynamic programming

Loop from **edDistDp**:

```
for i in xrange(1, len(x)+1):
    for j in xrange(1, len(y)+1):
        delt = 1 if x[i-1] != y[j-1] else 0
        D[i, j] = min(D[i-1, j-1]+delt, D[i-1, j]+1, D[i, j-1]+1)
```

	ϵ	G	C	T	A	T	G	C	C	A	C	G	C
ϵ	0	1	2	3	4	5	6	7	8	9	10	11	12
G	1												
C	2												
G	3												
T	4												
A	5						etc						
T	6												
G	7												
C	8												
A	9												
C	10												
G	11												
C	12												

Could we have filled the cells in a different order?

Edit distance: dynamic programming

Switched

```
for j in xrange(1, len(y)+1):
    for i in xrange(1, len(x)+1):
        delt = 1 if x[i-1] != y[j-1] else 0
        D[i, j] = min(D[i-1, j-1]+delt, D[i-1, j]+1, D[i, j-1]+1)
```

	ϵ	G	C	T	A	T	G	C	C	A	C	G	C
ϵ	0	1	2	3	4	5	6	7	8	9	10	11	12
G	1												
C	2												
G	3												
T	4												
A	5												
T	6					etc							
G	7												
C	8												
A	9												
C	10												
G	11												
C	12												

Yes: e.g. invert the loops

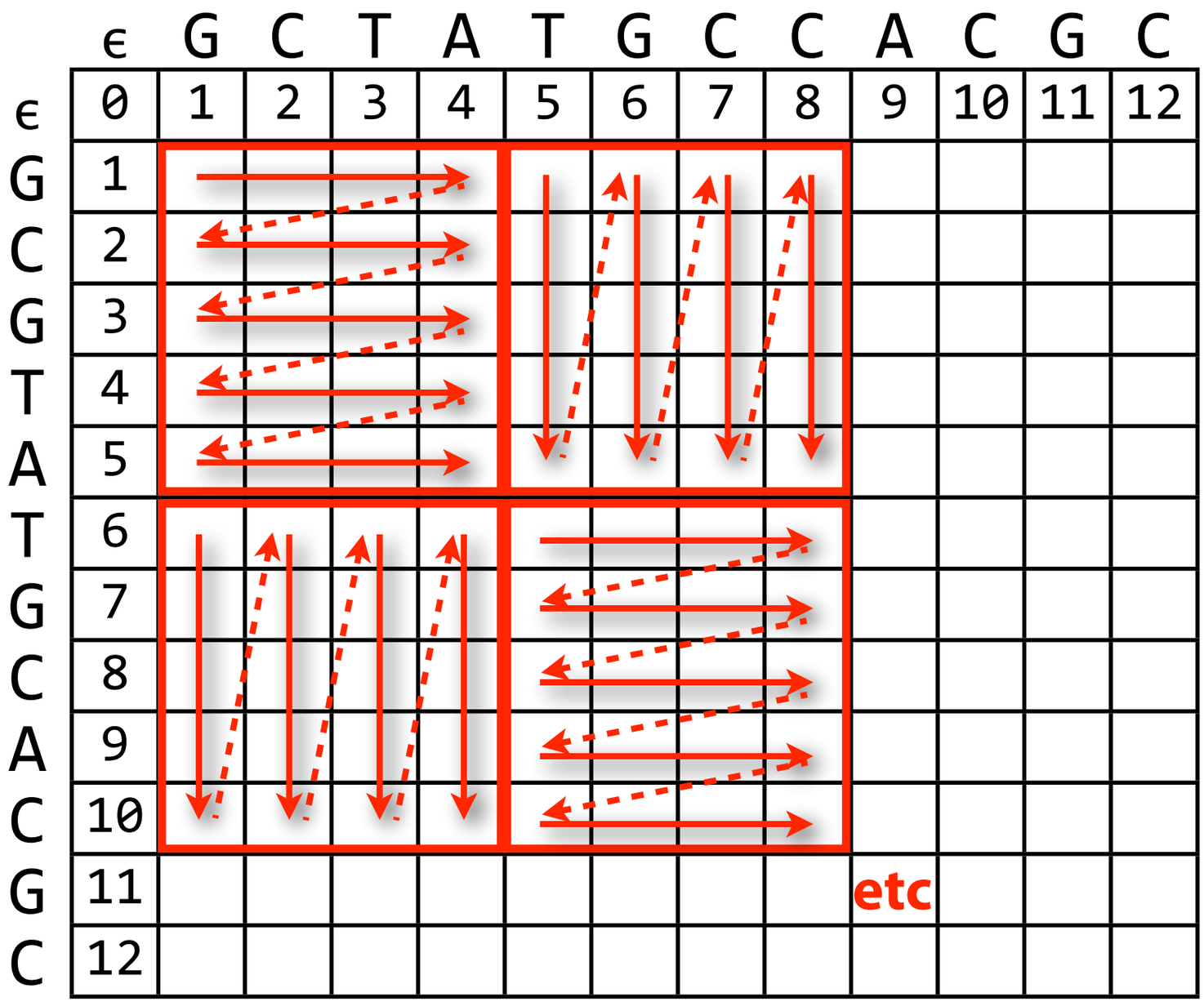
JOHNS HOPKINS
WHITING SCHOOL of ENGINEERING

Edit distance: dynamic programming

	ϵ	G	C	T	A	T	G	C	C	A	C	G	C
ϵ	0	1	2	3	4	5	6	7	8	9	10	11	12
G	1												
C	2												
G	3												
T	4				etc								
A	5												
T	6												
G	7												
C	8												
A	9												
C	10												
G	11												
C	12												

Or by anti-diagonal

Edit distance: dynamic programming



Or by submatrices

Longest common subsequence

Can we find length of the longest *subsequence* common to two strings?

A subsequence of *S* is a sequence of characters from *S* in the same relative order as in *S*, but not necessarily consecutive

	Some subsequences of *S*	*Not* subsequences of *S*
S: T A G T C A	T A C T T C A A T A T C A	G G T C G A T G

also a substring of *S*

Can we solve this similarly to how we solved edit distance?

Longest common subsequence

Weigh matches, mismatches and gaps such that (a) match is preferred to either mismatch or gap, and (b) consecutive gaps (insertion-then-deletion or deletion-then-insertion) are preferred to mismatch

$s(a, b)$:

Scoring function

	A	C	G	T	-
A	-1	1	1	1	0
C	1	-1	1	1	0
G	1	1	-1	1	0
T	1	1	1	-1	0
-	0	0	0	0	

Longest common subsequence

```
for i in xrange(1, len(x)+1):
    for j in xrange(1, len(y)+1):
        D[i, j] = min(D[i-1, j-1] + s(x[i-1], y[j-1]), # diagonal
                      D[i-1, j  ] + s(x[i-1], '-'),    # vertical
                      D[i  , j-1] + s('-',    y[j-1])) # horizontal
```

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

$s(a, b)$

	A	C	G	T	-
A	-1	1	1	1	0
C	1	-1	1	1	0
G	1	1	-1	1	0
T	1	1	1	-1	0
-	0	0	0	0	

-(-6) = 6 is LCS length

JOHNS HOPKINS
WHITING SCHOOL of ENGINEERING

Longest common subsequence

Backtrace works as usual in $O(m + n)$ steps

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

Diagonal moves give LCS:

A G T C A G

JOHNS HOPKINS
WHITING SCHOOL of ENGINEERING

Approximate matching

Can we search for the occurrence of *P* in *T* with the least edits?

T

		ϵ	A	A	C	C	C	T	A	T	G	T	C	A	T	G	C	C	T	T	G	G	A
	ϵ																						
	T																						
	A																						
	C																						
	G																						
P	T																						
	C																						
	A																						
	G																						
	C																						

Approximate matching

Can we search for the occurrence of *P* in *T* with least edits?

First idea: initialize first row with 0's rather than increasing integers

T

		ϵ	A	A	C	C	C	T	A	T	G	T	C	A	T	G	C	C	T	T	G	G	A
	ϵ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	T	1																					
	A	2																					
	C	3																					
	G	4																					
P	T	5																					
	C	6																					
	A	7																					
	G	8																					
	C	9																					

Approximate matching

Fill in the matrix with the usual edit-distance recurrence

D[*i*, *j*] equals the optimal edit distance between the length-*i* prefix of *P* and a substring of *T* ending at position *j*.

T

		ϵ	A	A	C	C	C	T	A	T	G	T	C	A	T	G	C	C	T	T	G	G	A
	ϵ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	T	1	1	1	1	1	1	0	1	0	1	0	1	1	0	1	1	1	0	0	1	1	1
	A	2	1	1	2	2	2	1	0	1	1	1	1	1	1	1	2	2	1	1	1	2	1
	C	3	2	2	1	2	2	2	1	1	2	2	1	2	2	2	1	2	2	2	2	2	2
	G	4	3	3	2	2	3	3	2	2	1	2	2	2	3	2	2	2	3	3	2	2	3
P	T	5	4	4	3	3	3	3	3	2	2	1	2	3	2	3	3	3	2	3	3	3	3
	C	6	5	5	4	3	3	4	4	3	3	2	1	2	3	3	3	3	3	3	4	4	4
	A	7	6	5	5	4	4	4	4	4	4	3	2	1	2	3	4	4	4	4	4	5	4
	G	8	7	6	6	5	5	5	5	5	4	4	3	2	2	2	3	4	5	5	4	4	5
	C	9	8	7	6	6	5	6	6	6	5	5	4	3	3	3	2	3	4	5	5	5	5

Approximate matching

Second idea: Pick lowest edit distance *in the bottom row*, backtrace from there

Can also find for all occurrences of *P* in *T* with ≤ *k* edits

T

		ϵ	A	A	C	C	C	T	A	T	G	T	C	A	T	G	C	C	T	T	G	G	A
	ϵ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	T	1	1	1	1	1	1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1
	A	2	1	1	2	2	2	1	0	1	2	2	2	2	2	2	2	2	2	2	2	2	2
	C	3	2	2	1	2	2	2	1	1	2	2	1	2	2	2	1	2	2	2	2	2	2
	G	4	3	3	2	2	3	3	2	2	1	2	2	2	3	2	2	2	3	3	2	2	3
P	T	5	4	4	3	3	3	3	3	2	2	1	2	3	2	3	3	3	2	3	3	3	3
	C	6	5	5	4	3	3	4	4	3	3	2	1	2	3	3	3	3	3	3	4	4	4
	A	7	6	5	5	4	4	4	4	4	4	3	2	1	2	3	4	4	4	4	4	5	4
	G	8	7	6	6	5	5	5	5	5	4	4	3	2	2	2	3	4	5	5	4	4	5
	C	9	8	7	6	6	5	6	6	6	5	5	4	3	3	3	2	3	4	5	5	5	5

```
T:  A A C C C T A T G T C A T G C C T T G G A
            | |   | | | |   | |
P:          T A C G T C A - G C
```