

GENOME ANALYSIS & BIOINFORMATICS

Week 3:

How can we align two
sequences?

DNA Sequence Alignment I: Motivation

You are studying a recently discovered human non-coding RNA.

You search it against the mouse genome using BLASTN (N for nucleotide) and obtain the following alignment:

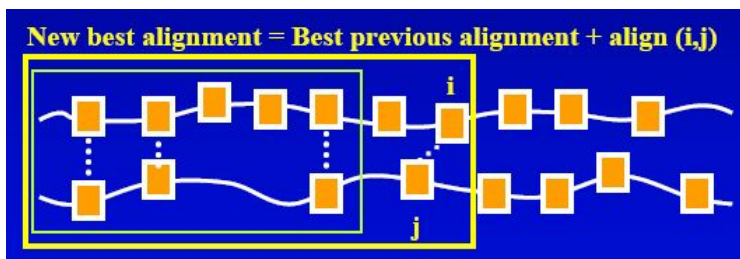
```
Q: 1   ttgacctagatgagatgtcgttcacttttactcaggtacagaaaa 45
      |||| | ||||| ||||| | ||||| ||||| || ||||| |||||
S: 403 ttgatctagatgagatgccattcacttttactgagctacagaaaa 447
```

Is this alignment significant?

Is this likely to represent a homologous RNA?

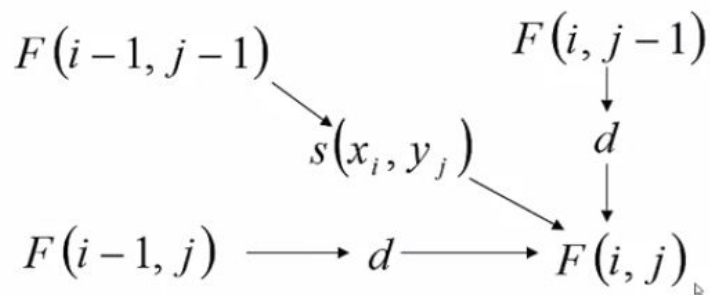
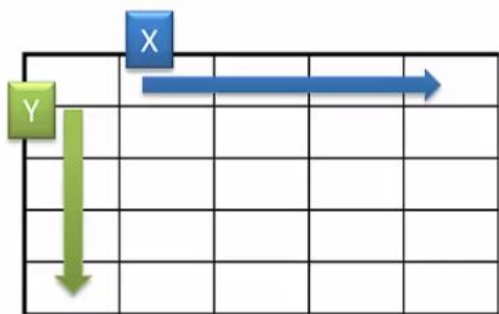
How to find alignments?

S	S	-
T	-	T



$$F(0,0) = 0$$

$$F(i, j) = \max \begin{cases} F(i-1, j-1) + s(x_i, y_j) & \mathbf{x_i \text{ aligned to } y_j} \\ F(i-1, j) + d & \mathbf{x_i \text{ aligned to a gap}} \\ F(i, j-1) + d & \mathbf{y_j \text{ aligned to a gap}} \end{cases}$$



EF-Tu and eEF1-alpha



eRF-3



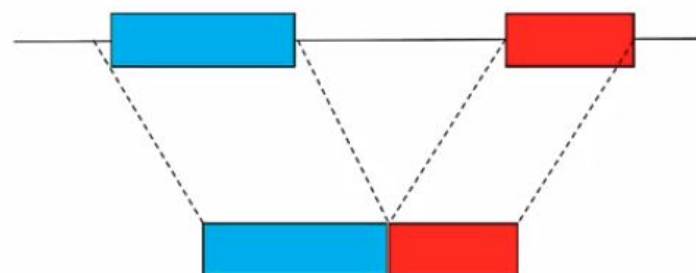
eIF-2gamma



IF-2 and eIF-5b



EF-G and eEF-2



Local alignment: example

Scoring

Match: +2

Mismatch: -1

Indel: -2

C T - A A

C T C A A

		0	1	2	3	4	5	6	7	8	9	10
		-	G	G	C	T	C	A	A	T	C	A
0	-	0	0	0	0	0	0	0	0	0	0	0
1	A	0	0	0	0	0	0	2	2	0	0	2
2	C	0	0	0	2	0	2	0	1	1	2	0
3	C	0	0	0	2	1	2	1	0	0	3	1
4	T	0	0	0	0	4	2	1	0	2	1	2
5	A	0	0	0	0	2	3	4	3	1	1	3
6	A	0	0	0	0	0	1	5	6	4	2	3
7	G	0	2	2	0	0	0	3	4	5	3	1
8	G	0	2	4	2	0	0	1	2	3	4	2

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

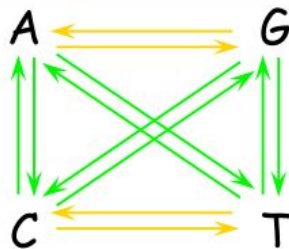
Identity

	A	T	C	G
A	5	-4	-4	-4
T	-4	5	-4	-4
C	-4	-4	5	-4
G	-4	-4	-4	5

BLAST

	A	T	C	G
A	0	5	5	1
T	5	0	1	5
C	5	1	0	5
G	1	5	5	0

Transition/
Transversion

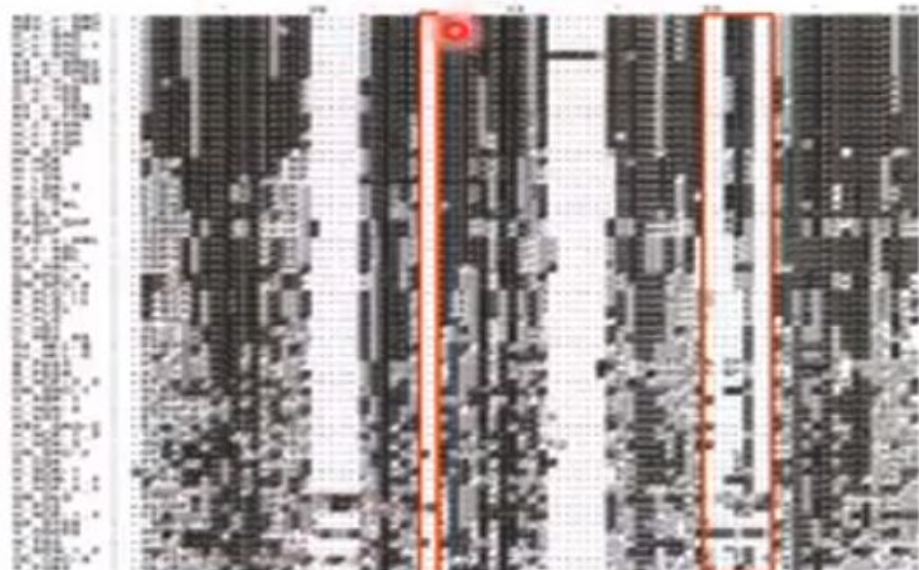
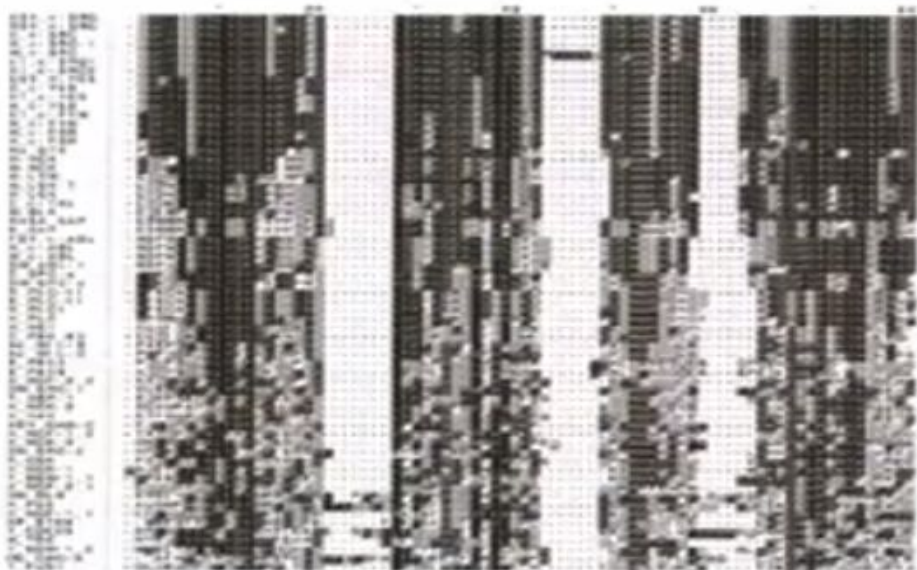


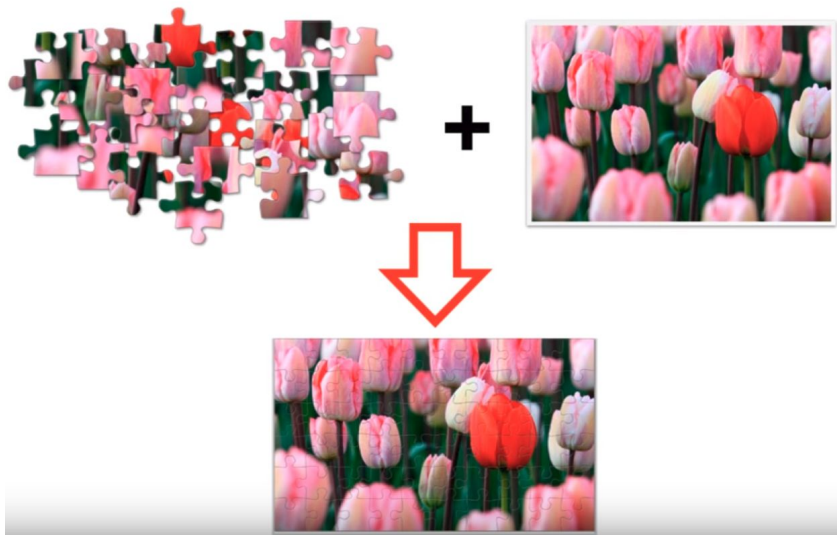
BLOSUM-62 matrix

C	9	small and polar residues																				
S	-1	4																				
T	-1	1	5																			
P	-3	-1	-1	7																		
A	0	1	0	-1	4	small and nonpolar																
G	-3	0	-2	-2	0	6																
N	-3	1	0	-2	-2	0	6	polar or acidic residues														
D	-3	0	-1	-1	-2	-1	1	6														
E	-4	0	-1	-1	-1	-2	0	2	5													
Q	-3	0	-1	-1	-1	-2	0	0	2	5												
H	-3	-1	-2	-2	-2	-2	1	-1	0	0	8	basic										
R	-3	-1	-1	-2	-1	-2	0	-2	0	1	0	5										
K	-3	0	-1	-1	-1	-2	0	-1	1	1	-1	2	5									
M	-1	-1	-1	-2	-1	-3	-2	-3	-2	0	-2	-1	-1	5	large and hydrophobic							
I	-1	-2	-1	-3	-1	-4	-3	-3	-3	-3	-3	-3	-3	1	4							
L	-1	-2	-1	-3	-1	-4	-3	-4	-3	-2	-3	-2	-2	2	2	4						
V	-1	-2	0	-2	0	-3	-3	-3	-2	-2	-3	3	2	1	3	1	4					
F	-2	-2	-2	-4	-2	-3	-3	-3	-3	-3	-1	-3	-3	0	0	0	-1	6	aromatic			
Y	-2	-2	-2	-3	-2	-3	-2	-3	-2	-1	2	-2	-2	-1	-1	-1	-1	3	7			
W	-2	-3	-2	-4	-3	-2	-4	-4	-3	-2	-2	-3	-3	-1	-3	-2	-3	1	2	11		
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W		

best parameters

default parameters





Read

CTCAAACTCCTGACCTTTGGTGATCCACCCGCCTAGGCCTTC x million

Reference

GATCACAGGTCATCACCTATTAACCACTCACGGGAGCTCTCCATGCATTTGGTATTTT
CGTCTGGGGGGATGCACGCGATAGCATTGCGAGACGCTGGAGCGGGAGACCCCTATGTC
GCAGTATCTGCTTTTGATTCTCTGCTCATCCTATTATTTATCGCACCTACGTTCAATATT
ACAGGGCAACATACTTACTAAAGTGTGTTAATTAATTAATGCTTGTAGGACATAATAATA
ACAATTGAATGTCTGCACAGCCACTTTCCACACAGACATCATAACAAAAATTTCCACCA
AACCCCCCTCCCCGCTTCTGGCCACAGCAATTTTCAAAATTTCTTTGGCGGTATGCAC
ACAAAGAACCTTAACACACAGCCTAACCAATTTTCAAAATTTCTTTGGCGGTATGCAC
TTTTAACAGTCACCCCACTAACCAATTTTCCCTCCCTCCTACTACTAAT
CTCATCAATACAAACCCCGCCCATTTACCCAGCACACACACCTTAACCCCATTA
CCCCGAACCAACCAACCCCAACCTTACCCCCACAGTTTATGTAGCTTCTCTCTCAAA
GCAATACACTGACCCGCTCAAACTCTGGATTTTGGATCCACCCAGCCTTTGGCTAAA
CTAGCCTTTCTATTAGCTCTTAGAAGATTACACATGCAAGCATCCCTCCAGTGAGT
TCACCTCTAAATCACCACGATCAAGGAACAAGCATCAAGCAGCAGTAATGCAGCTC
AAAACGCTTAGCCTAGCCACACCTCACGGGAAACAGCAGTGATTAAGCTTAGCAATAA
ACGAAAGTTTAACTAAGCTATACTTACCCAGGGTTGGTCAATTTCTTCCAGCCACCGC
GGTCACACGATTAACCAAGTCAATGAAGCCGGCGTAAAGAGTGTTATAGTACCCCC
TCCCCAATAAAGCTAAACTCACCTGTTGTAAAAAATCCAGTTCACAAATAGAC
TACGAAAGTGGCTTTAATATCTGAACCACTAGTAGTAAGCTTGGGATTAGA
TACCCCACTATGCTTAGCCCTAAACCTCAACAGTCAACCAAGTGGCAGAA
CACTACAGGCCACAGCTTAAACTCAAAGGACCTGGCGGTGCTTCATTTAGAGG
AGCCTGTTCTGTAATCGATAAACCCGATCAACCTCACCACCTCTTGCTATATA
CGGCCATCTTCAAGCAACCTGATGAAGGCTACAAAGTAAGCGCAAGTACGAGAG
ACGTTAGGTCAAGGTGAGCCATGAGGTGGCAAGAAATGGGCTACATTTTCT
AAAACCTACGATAGCCCTTATGAACCTTAAGGGTGAAGGTGGAATTTAGCAGTAA
AGTAGAGTGCTTAGTTGAACAGGGCCCTGAAGCGGTACACACCGCCGTCACCT
AGGTATACCTCAAAGGACATTTAACTAAAAACCCCTACGCATTATATAGAGGAGACA
CGTAACCTCAAACCTCTGCTTTGGTGATCCACCCGCCTTGGCTACCTGCATAATGAAG
AAGCACCAACTTACACTTAGGAGATTTCAACTTAACTTGACCGCTCTGAGCTAAACCTA
GCCCAAAACCCACTCACCTTACTACCAGACAACCTTAGCCAAACCTTTACCCAAATAA
AGTATAGGCGATAGAAATTGAAACCTGGCGCAATAGATATAGTACCGCAAGGGAAAGATG
AAAAATTATAACCAAGCATAATATAGCAAGGACTAACCCCTATACCTTCTGCATAATGAA
TTAACTAGAAATACTTTGCAAGGAGAGCCAAAGCTAAGACCCCCGAAACAGACGAGCT
ACCTAAGAACAGCTAAAAGAGCACACCCGTCTATGTAGCAAAATAGTGGGAAGTTTATA
GGTAGAGGCGACAACCTACCGAGCCTGGTGATAGCTGGTTGTCCAAGATAGAATCTTAG

x million

- nest site hunting, 482–87
- honeypot ants, *see Myrmecocystus*
- hormones, 106–9
 - see also* exocrine glands
- house (nest site) hunting, 482–92
- Hymenoptera (general), xvi
 - haplodiploid sex determination, 20–22
- Hypoponera* (ants), 194, 262, 324, 388
- inclusive fitness, 20–23, 29–42
- information measurement, 251–52
- intercastes, 388–89
 - see also* ergatogynes; ergatoid queens; gamergates
- Iridomyrmex* (ants), 266, 280, 288, 321
- Isoptera, *see* termites
- juvenile hormone, caste, 106–9, 372
- kin recognition, 293–98
- kin selection, 18–19, 23–24, 28–42, 299, 386
- Macrotermes* (termites), 59–60
- male recognition, 298
- mass communication, 62–63, 214–18
- mating, multiple, 155
- maze following, 119
- Megalomyrmex* (ants), 457
- Megaponera* (ants), *see Pachycondyla*
- Melipona* (stingless bees), 129
- Melophorus* (ants), repletes, 257
- memory, 117–19, 213
- Messor* (harvester ants), 212, 232
- mind, 117–19
- Monomorium*, 127, 212, 214, 216–17, 292
- motor displays, 235–47
- mound-building ants, 2
- multilevel selection, 7, 7–13, 24–29
- mutilation, ritual, 366–73
- mutualism, *see* symbioses, ants
- Myanmyrma* (fossil ants), 318
- Myopias* (ants), 326

G T G	0
T G C	1
G C G	2
C G T	3
G T G	4
T G T	5
G T G	6
T G G	7
G G G	8
G G G	9
G G G	10

T: G T G C G T G T G G G G G

C G T	3
G C G	2
G G G	8
G G G	9
G G G	10
G T G	0
G T G	4
G T G	6
T G C	1
T G G	7
T G T	5

T: G T G C G T G G G G G

P: G C G T G G

TGG > GTG

C G T	3
G C G	2
G G G	8
G G G	9
G G G	10
G T G	0
G T G	4
G T G	6
T G C	1
T G G	7
T G T	5

T: G T G C G T G G G G G

P: G C G T G G

After 1st
bisection

C G T	3
G C G	2
G G G	8
G G G	9
G G G	10
G T G	0
G T G	4
G T G	6
T G C	1
T G G	7
T G T	5

TGG > TGC

T: G T G C G T G G G G G

P: G C G T G G

After 2nd
bisection

C G T	3
G C G	2
G G G	8
G G G	9
G G G	10
G T G	0
G T G	4
G T G	6
T G C	1
T G G	7
T G T	5

TGG = TGG

T: G T G C G T G G G G G

P: G C G T G G

Suffix index

T: GTTATAGCTGATCGCGGCGTAGCGG
GTTATAGCTGATCGCGGCGTAGCGG
TTATAGCTGATCGCGGCGTAGCGG
TATAGCTGATCGCGGCGTAGCGG
ATAGCTGATCGCGGCGTAGCGG
TAGCTGATCGCGGCGTAGCGG
AGCTGATCGCGGCGTAGCGG
GCTGATCGCGGCGTAGCGG
CTGATCGCGGCGTAGCGG
TGATCGCGGCGTAGCGG
GATCGCGGCGTAGCGG
ATCGCGGCGTAGCGG
TCGCGGCGTAGCGG
CGCGGCGTAGCGG
GCGGCGTAGCGG
CGGCGTAGCGG
GGCGTAGCGG
GCGTAGCGG
CGTAGCGG
GTAGCGG
TAGCGG
AGCGG
GCGG
CGG
GG
G

Suffix index

T: GTTATAGCTGATCGCGGCGTAGCGG
GTTATAGCTGATCGCGGCGTAGCGG
TTATAGCTGATCGCGGCGTAGCGG
TATAGCTGATCGCGGCGTAGCGG
ATAGCTGATCGCGGCGTAGCGG
TAGCTGATCGCGGCGTAGCGG
AGCTGATCGCGGCGTAGCGG
GCTGATCGCGGCGTAGCGG
CTGATCGCGGCGTAGCGG
TGATCGCGGCGTAGCGG
GATCGCGGCGTAGCGG
ATCGCGGCGTAGCGG
TCGCGGCGTAGCGG
CGCGGCGTAGCGG
GCGGCGTAGCGG
CGGCGTAGCGG
GGCGTAGCGG
GCGTAGCGG
GTAGCGG
TAGCGG
AGCGG
GCGG
CGG
GG
G

$n(n+1)/2$
chars

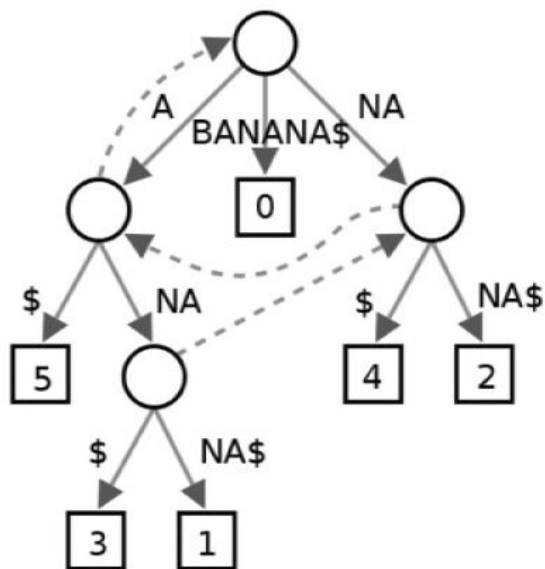
$T = \text{abaaba\$}$
 0123456

As with suffix tree,
 T is part of index

SA(T) =

6	\$
5	a \$
2	a a b a \$
3	a b a \$
0	a b a a b a \$
4	b a \$
1	b a a b a \$

m integers



Suffix tree
 ≥ 45 GB

6	\$
5	A\$
3	ANA\$
1	ANANA\$
0	BANANA\$
4	NA\$
2	NANA\$

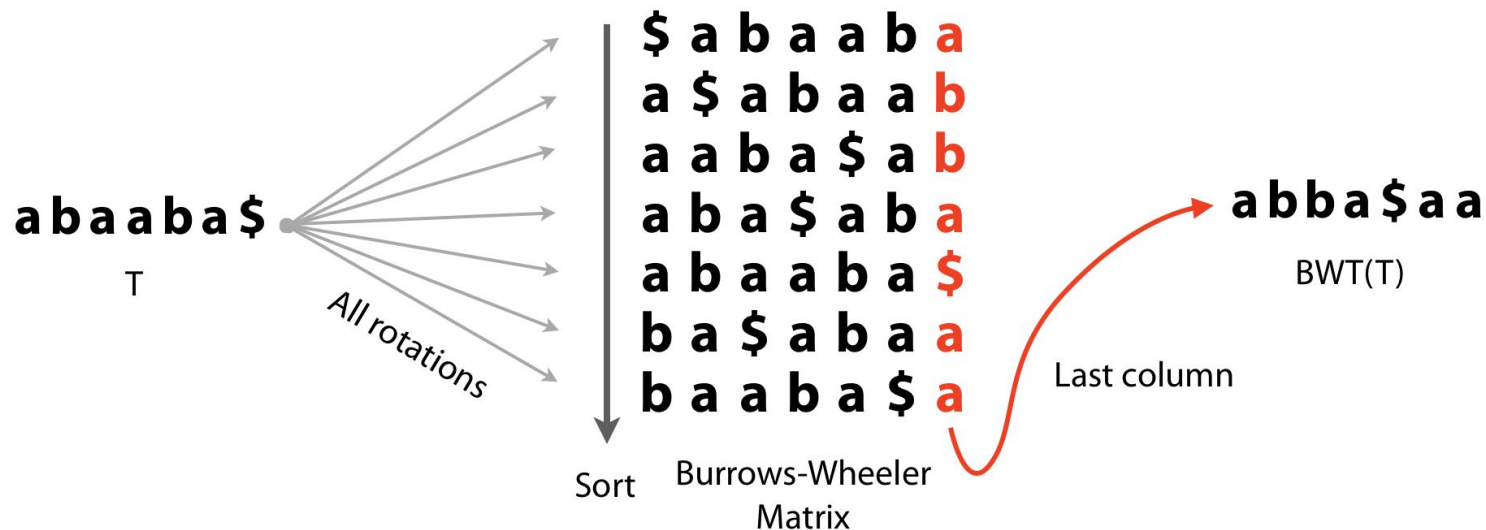
Suffix array
 ≥ 12 GB

\$ BANANA
A \$ BANAN
ANA \$ BAN
ANANA \$ B
BANANA \$
NA \$ BANA
NANA \$ BA

FM Index
 ~ 1 GB

Burrows-Wheeler Transform

Reversible permutation of the characters of a string, used originally for compression



How is it useful for compression?

How is it reversible?

How is it an index?

Burrows-Wheeler Transform: T-ranking

Give each character in T a rank, equal to # times the character occurred previously in T . Call this the *T-ranking*.

a₀ b₀ a₁ a₂ b₁ a₃ \$

Now let's re-write the BWM including ranks...

Burrows-Wheeler Transform

BWM with T-ranking:

	<i>F</i>						<i>L</i>
	\$	a ₀	b ₀	a ₁	a ₂	b ₁	a ₃
	a ₃	\$	a ₀	b ₀	a ₁	a ₂	b ₁
	a ₁	a ₂	b ₁	a ₃	\$	a ₀	b ₀
	a ₂	b ₁	a ₃	\$	a ₀	b ₀	a ₁
	a ₀	b ₀	a ₁	a ₂	b ₁	a ₃	\$
	b ₁	a ₃	\$	a ₀	b ₀	a ₁	a ₂
	b ₀	a ₁	a ₂	b ₁	a ₃	\$	a ₀

Look at first and last columns, called *F* and *L*

And look at just the **a**s

as occur in the same order in *F* and *L*. As we look down columns, in both cases we see: **a₃, a₁, a₂, a₀**

Burrows-Wheeler Transform

BWM with T-ranking:

<i>F</i>							<i>L</i>
\$	a ₀	b ₀	a ₁	a ₂	b ₁	a ₃	
a ₃	\$	a ₀	b ₀	a ₁	a ₂	b₁	
a ₁	a ₂	b ₁	a ₃	\$	a ₀	b₀	
a ₂	b ₁	a ₃	\$	a ₀	b ₀	a ₁	
a ₀	b ₀	a ₁	a ₂	b ₁	a ₃	\$	
b₁	a ₃	\$	a ₀	b ₀	a ₁	a ₂	
b₀	a ₁	a ₂	b ₁	a ₃	\$	a ₀	

Same with **b**s: **b₁**, **b₀**

Burrows-Wheeler Transform: LF Mapping

BWM with T-ranking:

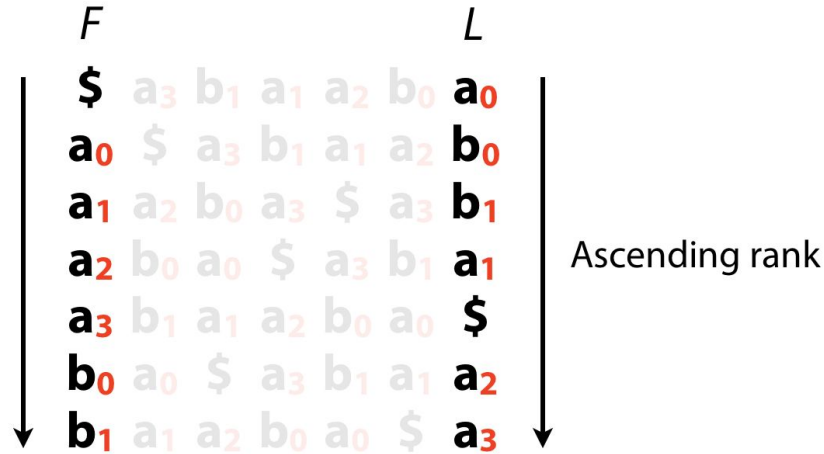
F	L
\$ a ₀ b ₀ a ₁ a ₂ b ₁ a ₃	
a ₃ \$ a ₀ b ₀ a ₁ a ₂ b ₁	
a ₁ a ₂ b ₁ a ₃ \$ a ₀ b ₀	
a ₂ b ₁ a ₃ \$ a ₀ b ₀ a ₁	
a ₀ b ₀ a ₁ a ₂ b ₁ a ₃ \$	
b ₁ a ₃ \$ a ₀ b ₀ a ₁ a ₂	
b ₀ a ₁ a ₂ b ₁ a ₃ \$ a ₀	

LF Mapping: The i^{th} occurrence of a character c in L and the i^{th} occurrence of c in F correspond to the *same* occurrence in T

However we rank occurrences of c , ranks appear in the same order in F and L

Burrows-Wheeler Transform: LF Mapping

BWM with B-ranking:



F now has very simple structure: a \$, a block of **a**s with ascending ranks, a block of **b**s with ascending ranks

Burrows-Wheeler Transform: reversing

Reverse BWT(T) starting at right-hand-side of T and moving left

Start in first row. F must have $\$$. L contains character just **prior** to $\$$: **a₀**

a₀: LF Mapping says this is same occurrence of **a** as first **a** in F . **Jump** to row *beginning* with **a₀**. L contains character just **prior** to **a₀**: **b₀**.

Repeat for **b₀**, get **a₂**

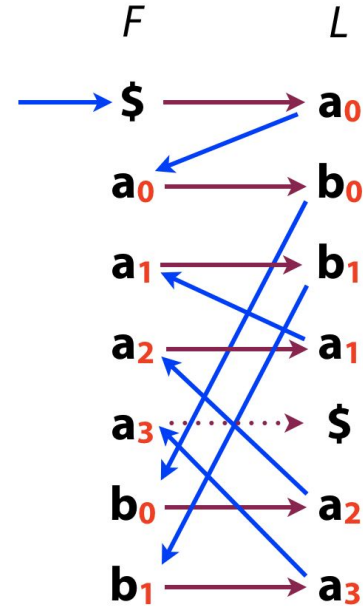
Repeat for **a₂**, get **a₁**

Repeat for **a₁**, get **b₁**

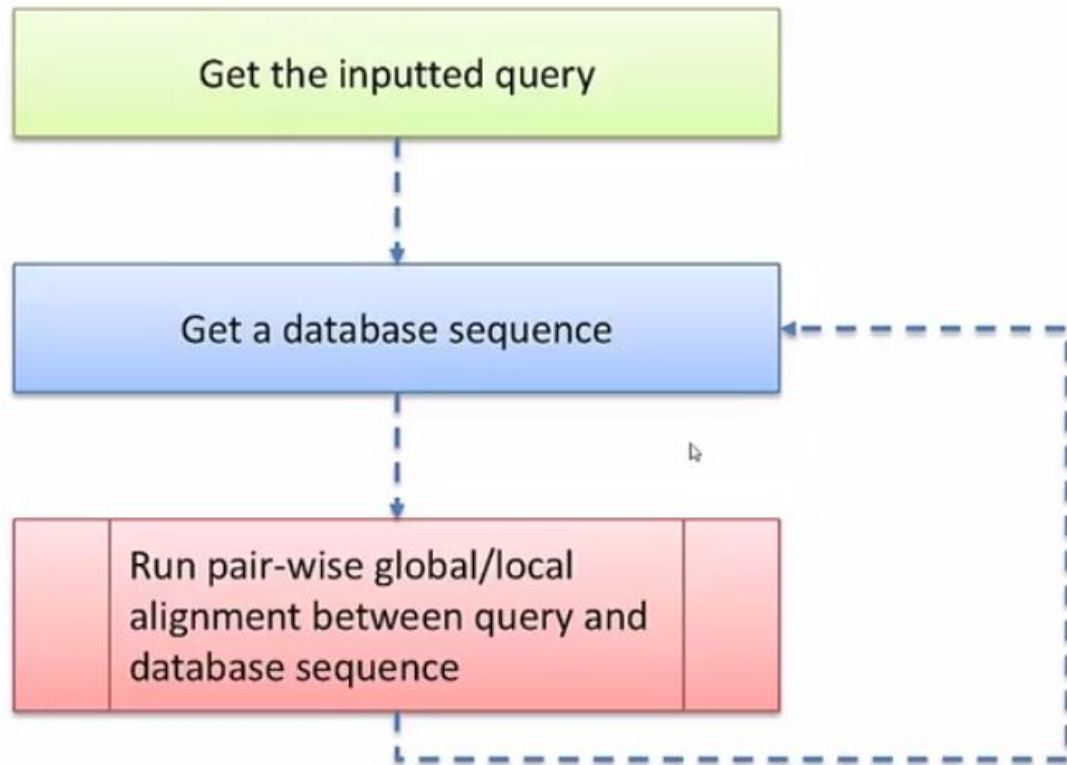
Repeat for **b₁**, get **a₃**

Repeat for **a₃**, get $\$$, done

Reverse of chars we visited = **a₃ b₁ a₁ a₂ b₀ a₀ \$** = T



A (naïve) algorithm for database searching



There are nm entries in the matrix.

Sequence X of length m

Sequence Y of length n

Dynamic programming matrix

Each entry requires a constant number c of operation(s).

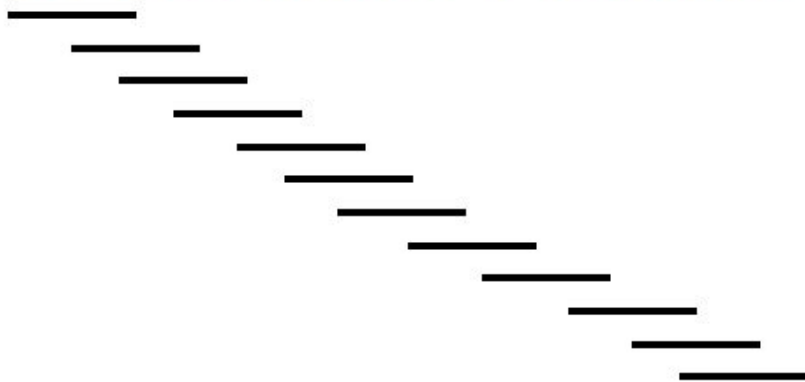
$c * m * n$ operations needed in total, for one pair-wise alignment.

Seeding

For a given **word length w** (usually 3 for proteins and 11 for nucleotides), slicing the query sequence into multiple continuous “**seed words**”

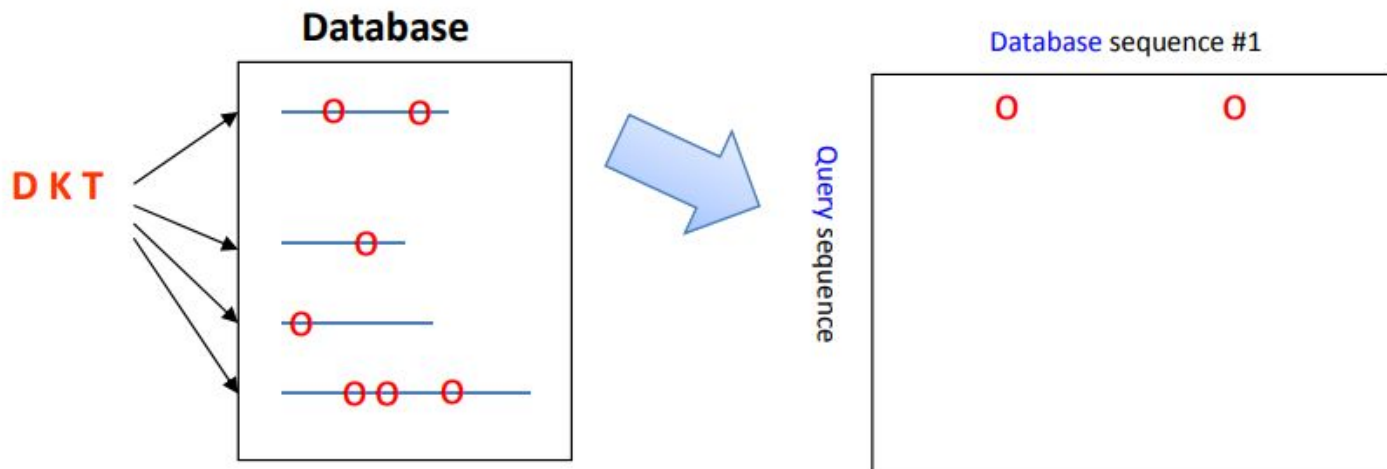
Query Sequence

M V L S P A D K T N V K A A W

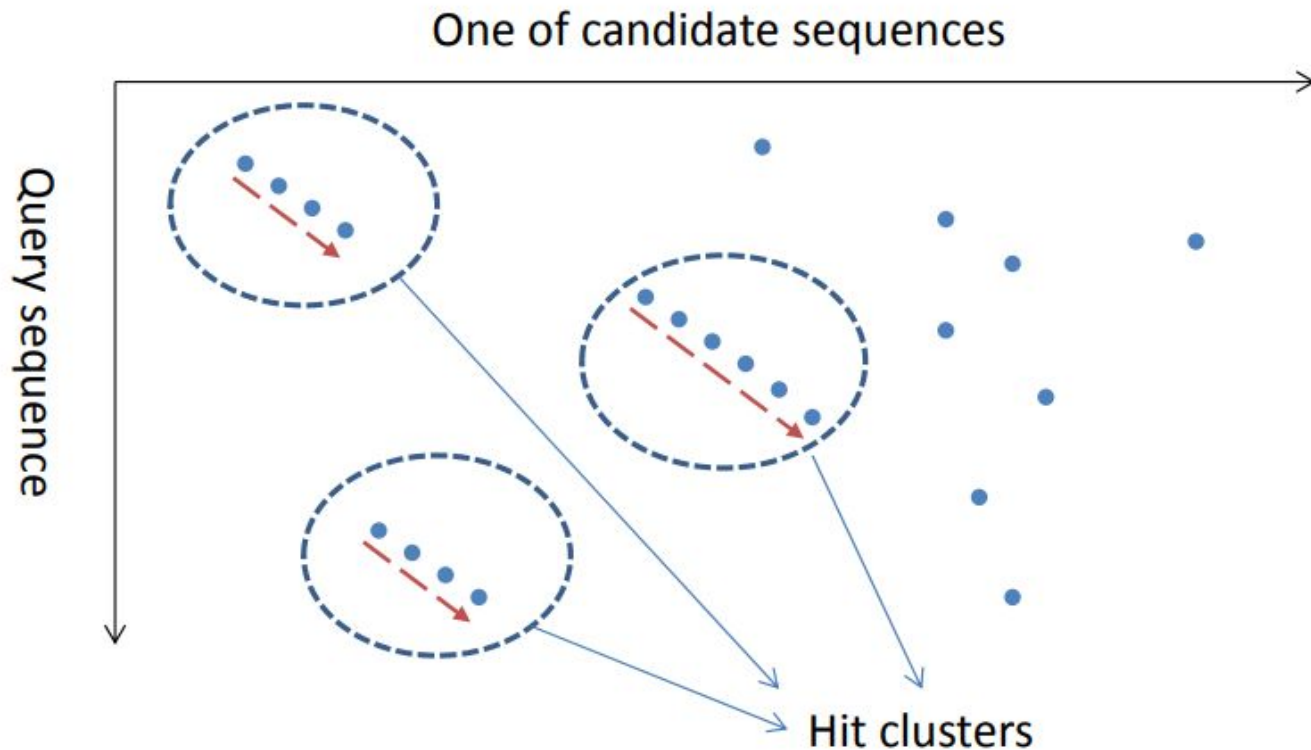


Speedup: Index database

The database was pre-indexed to quickly locate all positions in the database for a given seed.

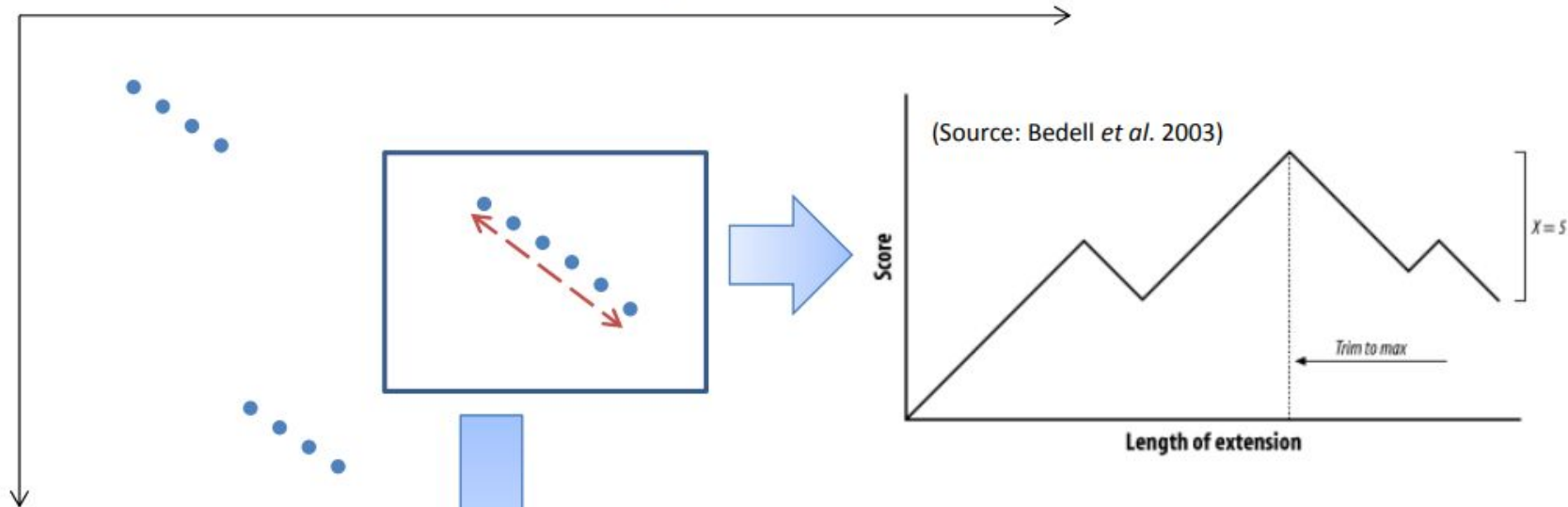


Diagonal *and* Two-hits



One of candidate sequences

Query sequence



$$F(0, 0) = 0$$

$$F(i, j) = \max \begin{cases} F(i-1, j-1) + s(x_i, y_j) \\ F(i-1, j) + d \\ F(i, j-1) + d \\ 0 \end{cases}$$

Quality Assessment

Given the large data volume, it's critical to provide some measures for assessing the **statistical significance** of a given hit.

