

# TOPIC 4: Sequence alignment

# Learning Outcomes

- Be able to define the two main methods of alignment.
- Understand the two main algorithms for NGS alignment, including strengths and weaknesses.
- Be able to read SAM format

# Sequence alignment

- Sequence alignment is a way of arranging the sequences of DNA, RNA, or protein to identify regions of similarity that may be a consequence of functional, structural, or evolutionary relationships between the sequences.

```
Q5E940_BOVIN -----MPREDRATWKSNYFLKIIQLDDYPKCFIVGADNVGSKMQIRMSLRGK-AVVLVGKNTMMRKAIRGHLENN--PALE 76
RLA0_HUMAN -----MPREDRATWKSNYFLKIIQLDDYPKCFIVGADNVGSKMQIRMSLRGK-AVVLVGKNTMMRKAIRGHLENN--PALE 76
RLA0_MOUSE -----MPREDRATWKSNYFLKIIQLDDYPKCFIVGADNVGSKMQIRMSLRGK-AVVLVGKNTMMRKAIRGHLENN--PALE 76
RLA0_RAT -----MPREDRATWKSNYFLKIIQLDDYPKCFIVGADNVGSKMQIRMSLRGK-AVVLVGKNTMMRKAIRGHLENN--PALE 76
RLA0_CHICK -----MPREDRATWKSNYFLKIIQLDDYPKCFIVGADNVGSKMQIRMSLRGK-AVVLVGKNTMMRKAIRGHLENN--PALE 76
RLA0_RANSY -----MPREDRATWKSNYFLKIIQLDDYPKCFIVGADNVGSKMQIRMSLRGK-AVVLVGKNTMMRKAIRGHLENN--PALE 76
Q7ZUG3_BRARE -----MPREDRATWKSNYFLKIIQLDDYPKCFIVGADNVGSKMQIRMSLRGK-AVVLVGKNTMMRKAIRGHLENN--PALE 76
RLA0_ICTPU -----MPREDRATWKSNYFLKIIQLDDYPKCFIVGADNVGSKMQIRMSLRGK-AVVLVGKNTMMRKAIRGHLENN--PALE 76
RLA0_DROME -----MVRNKAAWKAQYFIKVVLFDFEFKCFIVGADNVGSKMQIRMSLRGK-AVVLVGKNTMMRKAIRGHLENN--PALE 76
RLA0_DICDI -----MSGAG-SKRKNVFIEKATKLFITTDKMIIVAEADVFSSQLQKIRKSIIRGI-CAVLVGKNTMMRKAIRGHLENN--PALE 75
Q54LP0_DICDI -----MSGAG-SKRKNVFIEKATKLFITTDKMIIVAEADVFSSQLQKIRKSIIRGI-CAVLVGKNTMMRKAIRGHLENN--PALE 75
RLA0_PLAF8 -----MAKLSQKKQMYIEKLSSLIQQYSKILIVHDNVGSKMQIRMSLRGK-AVVLVGKNTMMRKAIRGHLENN--PALE 76
RLA0_SULAC -----MIGLAVTTTKKIAKWKVDEVAELTCKLKTHTKIIIANIEGFPADKLHEIRKKLRGK-ADIKVTNNLFIKIAKKNAG--VDK 79
RLA0_SULTO -----MRIMAVITQERKIAKWKIEEVKELEKLEVTHTIIIANIEGFPADKLHEIRKKLRGK-AEIKVTNNLFIKIAKKNAG--LDVS 80
RLA0_SULSO -----MKRLALALKQKRVASWKEEVKELEKLEVTHTIIIANIEGFPADKLHEIRKKLRGK-ADIKVTNNLFIKIAKKNAG--LDVS 80
RLA0_AERPE MSVVSIVGQMYKREKPIPEWKTLMLELELFSKRRVFLADLTGTFVVRVYRKKLWKK-KPMVAKKRIILRAMKAAGLE--LDDN 86
RLA0_PYRAE MMLAIGKRRYVTRQVPARKVKIVSEATELLQKVPYVFLFDLHGLSSRIILHEVYRILRRY-GVIKIKIPFLFKIAFTKVYGG--IPAL 85
RLA0_METAC -----MAEERHHTEHIPQWKDEIENIKELIQSHKVFQMGVLEGILATKMKIRRDLDKV-AVLKVRNTLLEALNQLG--ETIP 78
RLA0_METMA -----MAEERHHTEHIPQWKDEIENIKELIQSHKVFQMGVLEGILATKMKIRRDLDKV-AVLKVRNTLLEALNQLG--ETIP 78
RLA0_ARCFU -----MAAVRCS--PDEYKVRRAVEEIKRMISSEKPVVAIVSFRNVPAGOMOKIRREFRGK-AEIKVTNNLFIKIAKKNAG--LDVS 75
RLA0_METKA MAVKAKGQPPSCYEKKVAEWKRRVEVKELKELMDEYENVGLVDLEGIPAPOLQEIIRAKLRERDIIIRMSNTLMRIALEEKLDER--PELE 88
RLA0_METTH -----MAHVAEWKKEVEQLHDLIKGVVGVGIANLADIPARQLQKMRQTLRDS-ALIRMSNTLMRIALEEKLDER--PELE 88
RLA0_METTL -----MITAESEHKIAPWKIEEVNKLKELLKNGQIVLVDMMEVPARQLQEIIRDKIR-ETMLKMSNTLMRIALEEKLDER--PELE 82
RLA0_METVA -----MIDAKSEHKIAPWKIEEVNKLKELLKNSANVIALIDMMEVPARQLQEIIRDKIR-ETMLKMSNTLMRIALEEKLDER--PELE 82
RLA0_METJA -----METKVKAHVADWKIEEVKTLKGLIKSKPVVAIVDMDVPAPOLEIRDKIR-DKVKLRMSNTLMRIALEEKLDER--PELE 81
RLA0_PYRAB -----MAHVAEWKKEVEELANLKSYPVIALVDVSSMPAYPLSQMRRLLIRENGGLRVSRNTLIELAIKKAAAGELCKPELE 77
RLA0_PYRHO -----MAHVAEWKKEVEELAKLKSYPVIALVDVSSMPAYPLSQMRRLLIRENGGLRVSRNTLIELAIKKAAAGELCKPELE 77
RLA0_PYRFU -----MAHVAEWKKEVEELANLKSYPVIALVDVSSMPAYPLSQMRRLLIRENGGLRVSRNTLIELAIKKAAAGELCKPELE 77
RLA0_PYRKO -----MAHVAEWKKEVEELANLKSYPVIALVDVSSMPAYPLSQMRRLLIRENGGLRVSRNTLIELAIKKAAAGELCKPELE 76
RLA0_HALMA MSSESEKRTETPEWKEQEVDAIVMIESYSESGVGVNAGIPERQLDMRRDLHGT-AELRVSRNTLIEALDDVD--DGLD 79
RLA0_HALVO MSSESEVQTEVTPQWKEEVDELVDVIESYSESGVGVNAGIPERQLDMRRDLHGT-AELRVSRNTLIEALDDVD--DGLD 79
RLA0_HALSA -----MSAEQRRTTEVPEWKEQEVDAIVMIESYSESGVGVNAGIPERQLDMRRDLHGT-AELRVSRNTLIEALDDVD--DGLD 79
RLA0_THAAC -----MKEVSQKKELVNEITRIKASRSVAIVDAGIRROIIDIRGKNRGK-INLVKIKKELLFKALENLGD--EKLS 72
RLA0_THIVO -----MRKINAKKKEIVSELAIDITKSAVAIVDVKVRRMODIRAKNRDK-VKIKVVKKELLFKALENLGD--EKLS 72
RLA0_PICTO -----MTEPAQWKIDFVKKLENEINSRKVAIVSILKRLNNPFIKIRNSIRDK-ARIKVRARLLRLAIENTGK--NNIV 72
ruler 1.....10.....20.....30.....40.....50.....60.....70.....80.....90
```

A multiple alignment of protein sequences

# Pairwise alignment

- Alignment of two sequences is a relatively straightforward computational problem, but...
  - there are many possible alignments
  - there can be a very large reference
- NOTE: Two sequences can always be aligned and there can be more than one optimal solution

# Methods of alignment

- By hand
- Mathematical approach
  - Dynamic programming (slow, but optimal)
- Heuristic methods (fast, but approximate)
  - BLAST, short read aligners

# Think-pair-share

How could you quantify the quality of these DNA alignments?

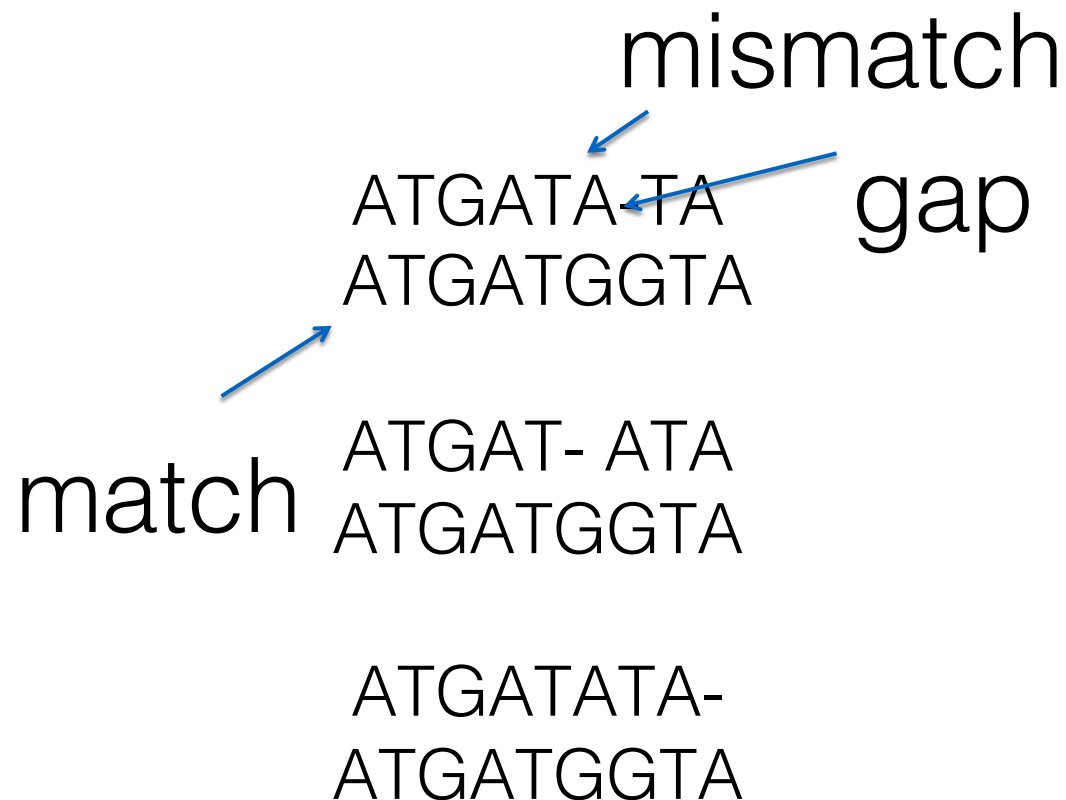
Example:

ATGATA-TA  
ATGATGGTA

ATGAT- ATA  
ATGATGGTA

ATGATATA-  
ATGATGGTA

**scoring matrix:** a table of values that describes the probability of a biologically meaningful amino-acid or nucleotide residue-pair occurring in an alignment.



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

Gap = -2

# Scoring methods

- Scoring systems:
  - Each symbol pairing is assigned a numerical value, based on a symbol comparison table.
    - nucleotides
    - amino acids (PAM, BLOSUM)
- Gap penalties:
  - Opening: The cost of introducing a gap.
  - Extension: The cost to elongate a gap.



# BLOSUM62

A	4																			
R	-1	5																		
N	-2	0	6																	
D	-2	-2	1	6																
C	0	-3	-3	-3	9															
Q	-1	1	0	0	-3	5														
E	-1	0	0	2	-4	2	5													
G	0	-2	0	-1	-3	-2	-2	6												
H	-2	0	1	-1	-3	0	0	-2	8											
I	-1	-3	-3	-3	-1	-3	-3	-4	-3	4										
L	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4									
K	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5								
M	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5							
F	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6						
P	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7					
S	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4				
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5			
W	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11		
Y	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7	
V	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4
X	0	-1	-1	-1	-2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-2	0	0	-2	-1	-1
	A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V

Positive for chemically similar substitution

Common amino acids have low weights

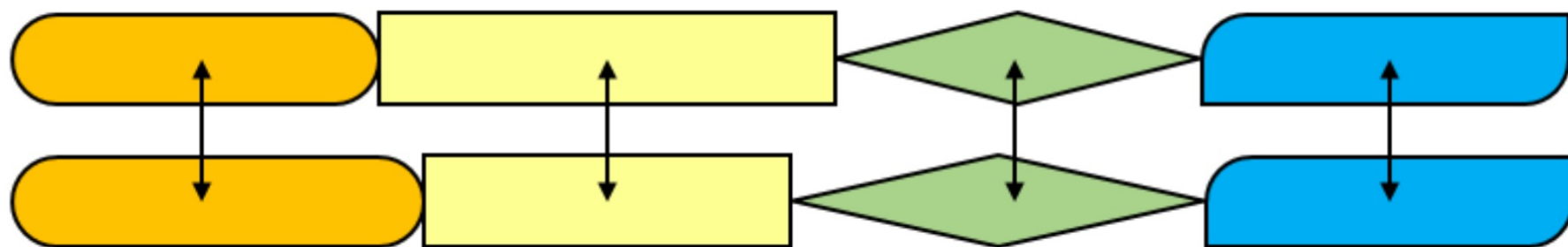
Rare amino acids have high weights

# Gap penalties

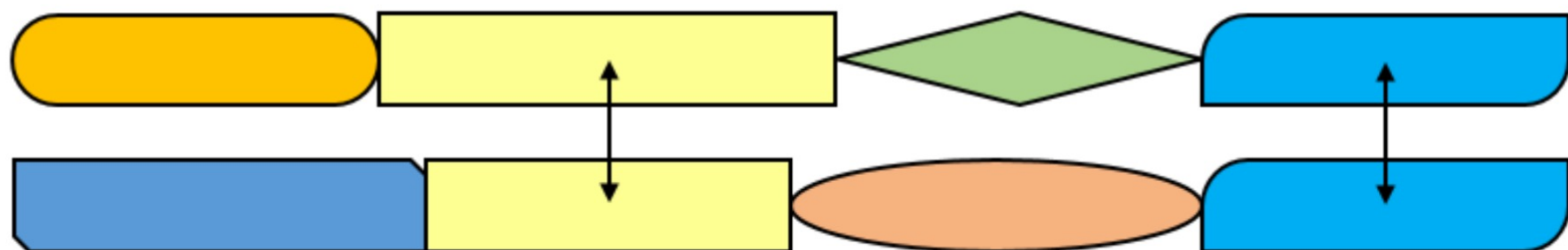
- Too little gap penalty gives nonsense non-homologous alignments.
- Gaps are common, so too high gap penalty removes real alignments.
- “Affine” gap penalty has a large penalty to introduce a gap and a smaller penalty to extend one.

# Dynamic programming

- Dynamic programming is a general programming technique.
- It structures a large search space into a succession of stages
  - The initial stage contains trivial solutions to sub-problems
  - Each partial solution in a later stage can be calculated by recurring a fixed number of partial solutions in an earlier stage
  - The final stage contains the overall solution



Global Alignment



Local Alignment

# Global vs Local alignments

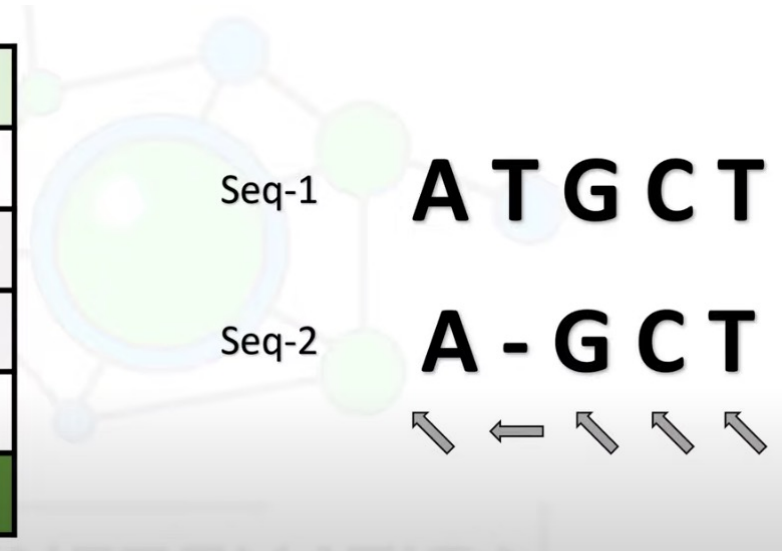
- Global alignment algorithms start at the beginning of two sequences and add gaps to each until the end of one is reached (Needleman-Wunsch).
- Local alignment algorithms find the region (or regions) of highest similarity between two sequences and build the alignment outward from there (Smith-Waterman).
- Demo of the two algorithms:  
<https://gtuckerkellogg.github.io/pairwise/demo/>
- Video demonstrating exactly how to solve the Needleman-Wunsch alignment by hand  
<https://www.youtube.com/watch?v=ipp-pNRlp4g>

# Basic principles of dynamic programming

There are too many comparisons to try them all so instead:

- Build alignment path matrix
- Stepwise calculation of score values
- Backtracking (evaluation of optimal path)

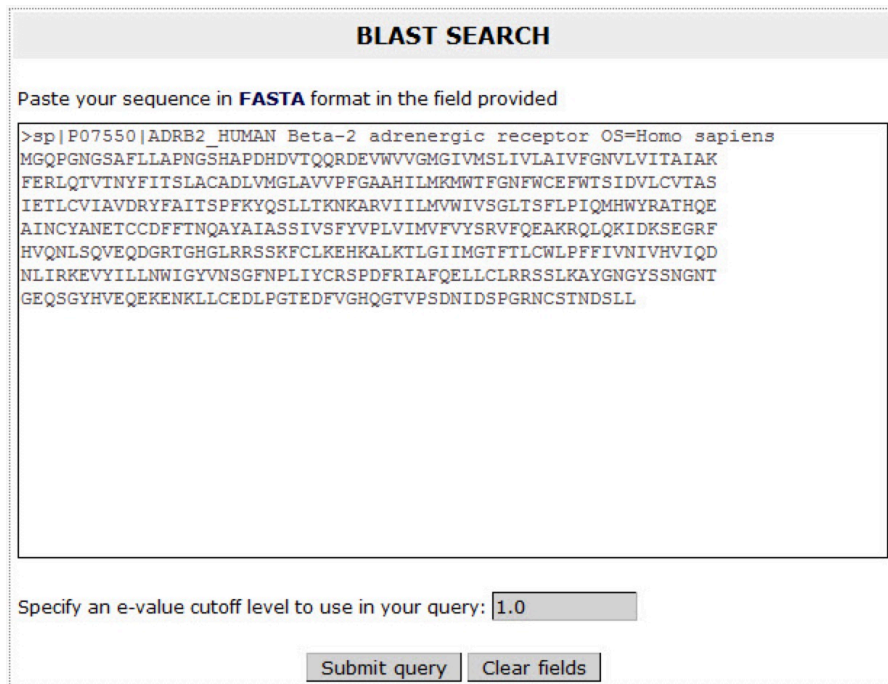
		A	T	G	C	T
	0	-2	-4	-6	-8	-10
A	-2	1	-1	-3	-5	-7
G	-4	-1	0	0	-2	-4
C	-6	-3	-2	-1	1	-1
T	-8	-5	-2	-3	-1	2



# Basic Local Alignment Search Tool (BLAST)

Used for finding alignments between a small number of sequences and a large database

query vs database



**BLAST SEARCH**

Paste your sequence in **FASTA** format in the field provided

```
>sp|P07550|ADRB2_HUMAN Beta-2 adrenergic receptor OS=Homo sapiens
MGQPGNGSAFLLAPNGSHAPDHDVTQQRDEVVVGMGIVMSLIVLAIVFGNVLVITAIK
FERLQTVTNFYFITSACADLVMGLAVVFFGAHILMKMWTFGNFWCFWTSIDVLCVTAS
IETLCVIAVDRYFAITSPFKYQSLLTKNKARVILMVWIVSGLTSFLPIQMHWRATHQE
AINCYNANETCCDFFFTNQAYAIASSIVSFYVPLVIMVFVYSRVFQEAQRQLQKIDKSEGRF
HVQNLSQVEQDGRGTGHGLRRSSKFCLEKHKALKTGLIIMGFTLCWLPFFIVNIVHVIQD
NLIRKEVYILLNWIGYVNSGFNPLIYCRSPDFRIAFQELLCLRRSSLKAYNGYSSNGNT
GEQSGYHVEQEKENKLLCEDLPGTEDFVGHQGTVPFSDNIDSPGRNCSTNDSL
```

Specify an e-value cutoff level to use in your query:

Examples:

nr = non-redundant

Swiss Prot

Model organisms (TAIR)

Your favorite sequences

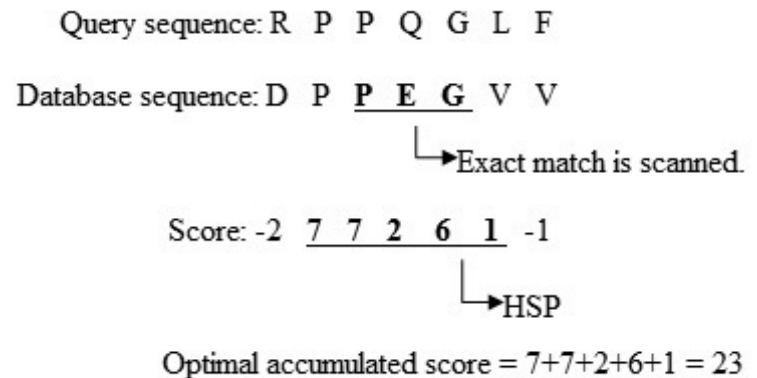
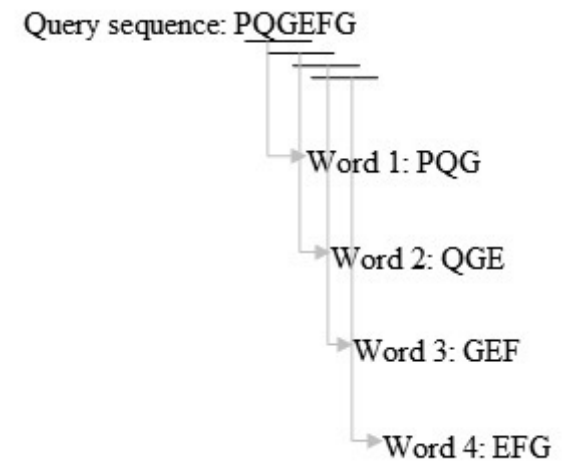
<http://www.ncbi.nlm.nih.gov/books/NBK21097/>

# BLAST

BLAST searches for regions of similarity and doesn't try to align the entire sequence (local alignment). A global alignment aligns two sequences over the entire length

Designed to identify homologous sequences.

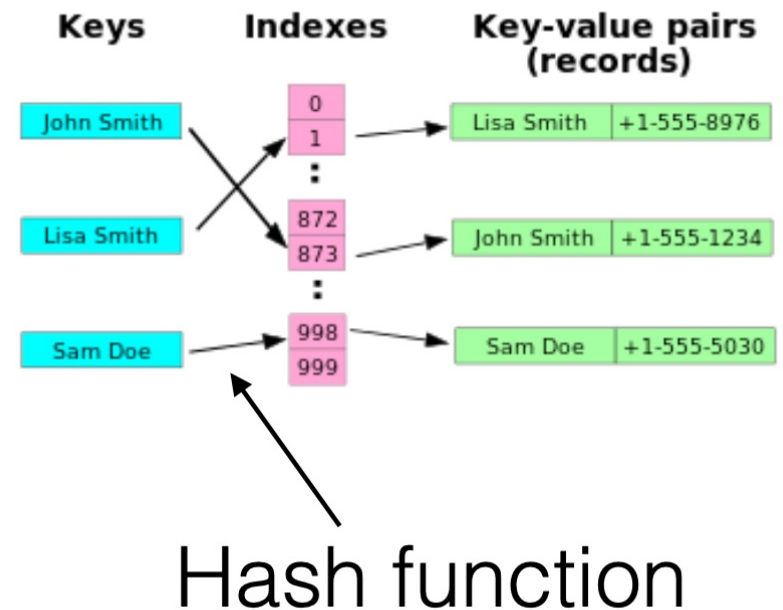
Hashed seed-extend algorithm





# Hashed seed-extend algorithm

- A “hash” is a structure used in computer programming
- It is a way of storing information in a look-up table
- Allows efficient searching



Step 1. Divide up your query sequence  
into **words** (typically 11 for DNA and 3 for  
proteins)

query sequence      LIAWHCMPNAAA

list of words      LIA  
                    IAW  
                    AWH  
                    WHC

Step 2. Find sequences in the database that have matches to query words with a score greater than your **threshold**

query sequence LIAWHCMPNAAA

query word WHC

database  
sequence 1 ...WHC...

database  
sequence 2 ....WIC...

database  
sequence 3 ...AIC...

# BLOSUM62

A	4																				
R	-1	5																			
N	-2	0	6																		
D	-2	-2	1	6																	
C	0	-3	-3	-3	9																
Q	-1	1	0	0	-3	5															
E	-1	0	0	2	-4	2	5														
G	0	-2	0	-1	-3	-2	-2	6													
H	-2	0	1	-1	-3	0	0	-2	8												
I	-1	-3	-3	-3	-1	-3	-3	-4	-3	4											
L	-1	-2	-3	-4	-1	-2	-3	-4	-3	2	4										
K	-1	2	0	-1	-3	1	1	-2	-1	-3	-2	5									
M	-1	-1	-2	-3	-1	0	-2	-3	-2	1	2	-1	5								
F	-2	-3	-3	-3	-2	-3	-3	-3	-1	0	0	-3	0	6							
P	-1	-2	-2	-1	-3	-1	-1	-2	-2	-3	-3	-1	-2	-4	7						
S	1	-1	1	0	-1	0	0	0	-1	-2	-2	0	-1	-2	-1	4					
T	0	-1	0	-1	-1	-1	-1	-2	-2	-1	-1	-1	-1	-2	-1	1	5				
W	-3	-3	-4	-4	-2	-2	-3	-2	-2	-3	-2	-3	-1	1	-4	-3	-2	11			
Y	-2	-2	-2	-3	-2	-1	-2	-3	2	-1	-1	-2	-1	3	-3	-2	-2	2	7		
V	0	-3	-3	-3	-1	-2	-2	-3	-3	3	1	-2	1	-1	-2	-2	0	-3	-1	4	
X	0	-1	-1	-1	-2	-1	-1	-1	-1	-1	-1	-1	-1	-1	-2	0	0	-2	-1	-1	-1
A	R	N	D	C	Q	E	G	H	I	L	K	M	F	P	S	T	W	Y	V	X	

Positive for chemically similar substitution

Common amino acids have low weights

Rare amino acids have high weights

What is the score for each of these alignments (use the BLOSUM62 scoring matrix)?

Step 3. If a match is found extend the alignment around it in each direction for as long as the cumulative score is above a threshold (e.g. 0) or if it falls off by more than X from its maximum. (or you get to the end of the sequence)

This extended match is called a **High Scoring Pair (HSP)**

query sequence           LIA**WHC**MPNAAA

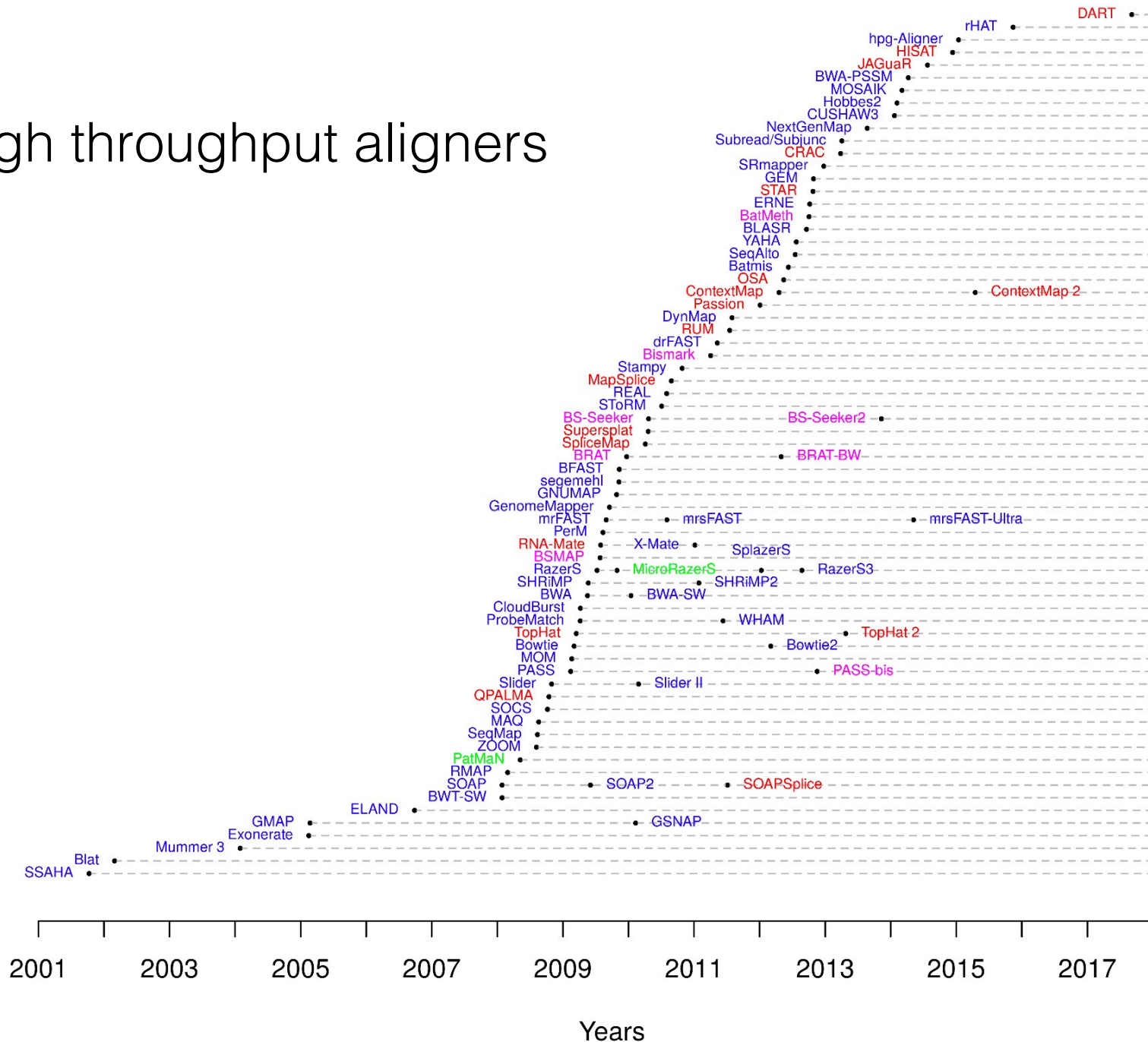
database sequence 1   IIA**WHC**MPNDDA



# BLAST

- Why not use BLAST for short read data (e.g., aligning short reads to a reference)?
- Typically takes 0.1 to 1 second to search 1 sequence against a database
- 60 million reads equates to 70 CPU days

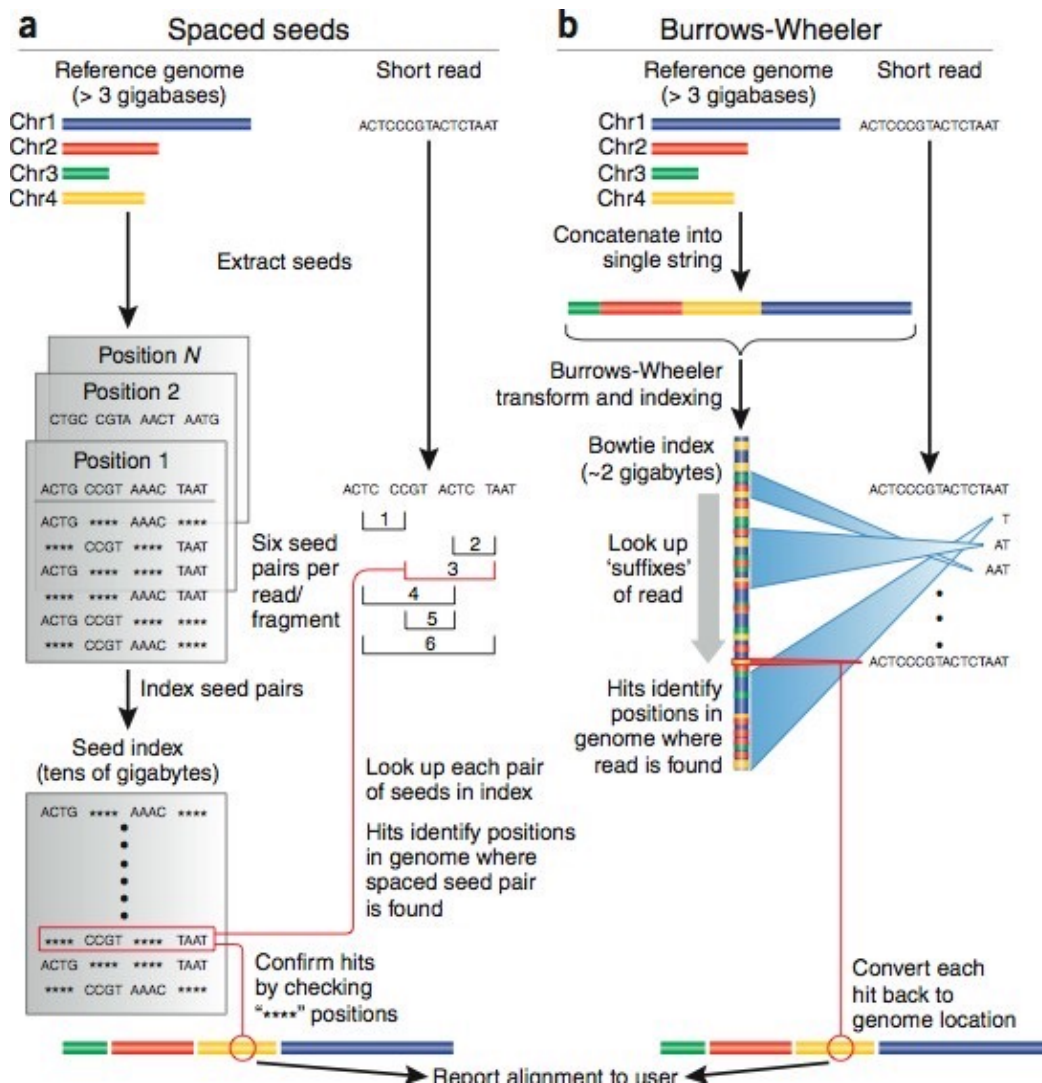
# High throughput aligners



# Short read alignment is hard

- Billions of short sequences aligned to a very long reference
- Short reads contain less information and are less likely to have a unique mapping location

# Approaches to align short reads



Trapnell &  
Salzberg 2009



# Hashed seed-extend algorithms

- Two step process:
  - Identify a match to the seed sequence in the reference
  - Extend match using sensitive (but slow) Smith-Waterman algorithm

# Seed-extend algorithm

Reference sequence:

...GATCTCGATCGATGATCGTAGGATTGATCAGCTA...

Short read:

TCGATCGATGATCGAAGGATTGATCAG

# Seed-extend algorithm

Reference sequence:

...GATCTCGATCGATGATCGTAGGATTGATCAGCTA...

Short read:

TCGATCGAT

9bp seed

GATCGAAGG

9bp seed

ATTGATCAG

9bp seed

The algorithm will try to match each seed to the reference.  
If there is a match with any seed, it performs a local alignment

# Seed-extend algorithm

Reference sequence:

seed      ->Extend with Smith-Waterman->

...GATCTCGATCGATGATCGTAGGATTGATCAGCTA...

TCGATCGATGATCGAAGGATTGATCAG

Short read:

TCGATCGAT

9bp seed

GATCGAAGG

9bp seed

ATTGATCAG

9bp seed

Here there is a match with at least one seed

# Seed-extend algorithm

Reference sequence:

...GATCTCGATCGATGATCGTAGGATTGATCAGCTA...

Short read:

TAGATCGAT

9bp seed

GATCGAAGG

9bp seed

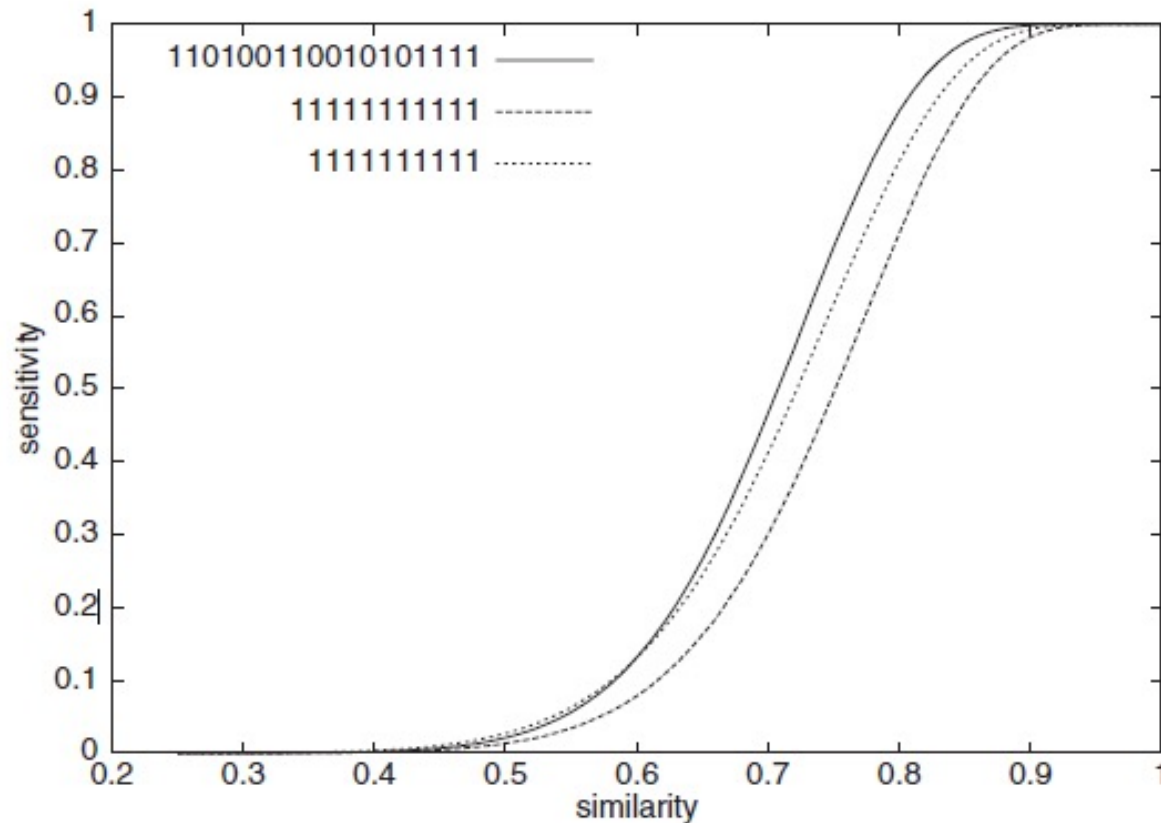
ATTGAGCAG

9bp seed

With three sequencing errors/SNPs, there can be no matches

# Spaced seeds

- To increase sensitivity we can use spaced-seeds:



# Spaced seeds

- To increase sensitivity we can use spaced-seeds:

111111111

Consecutive seed template with **length** 9bp

GATAGCTAGCTAAT

Reference

AGCTAGCTA

Query

10101101011011

Consecutive seed template with **weight** 9bp

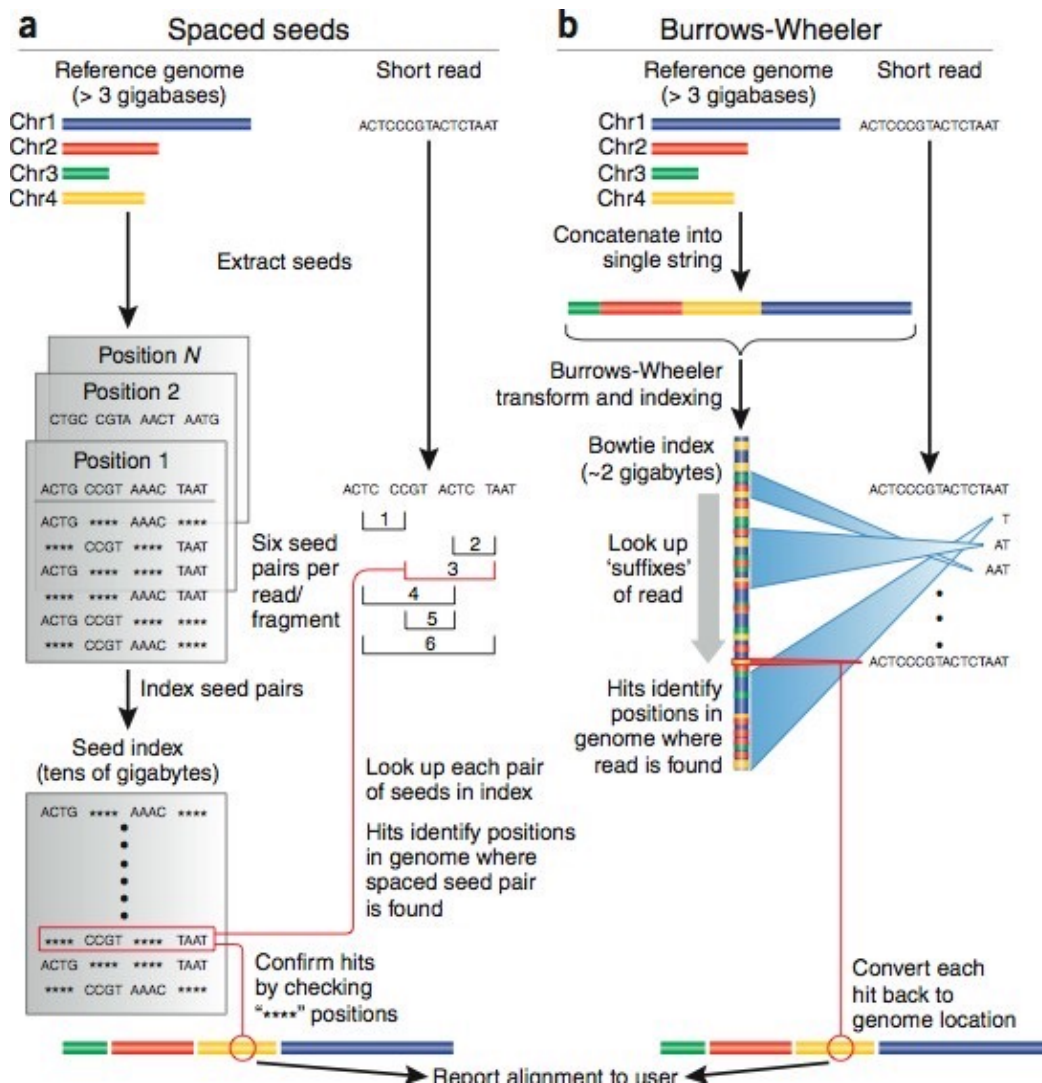
GATAGCTAGCTAAT

Reference

GATAGCGAGCTAAT

Query

# Approaches to align short reads

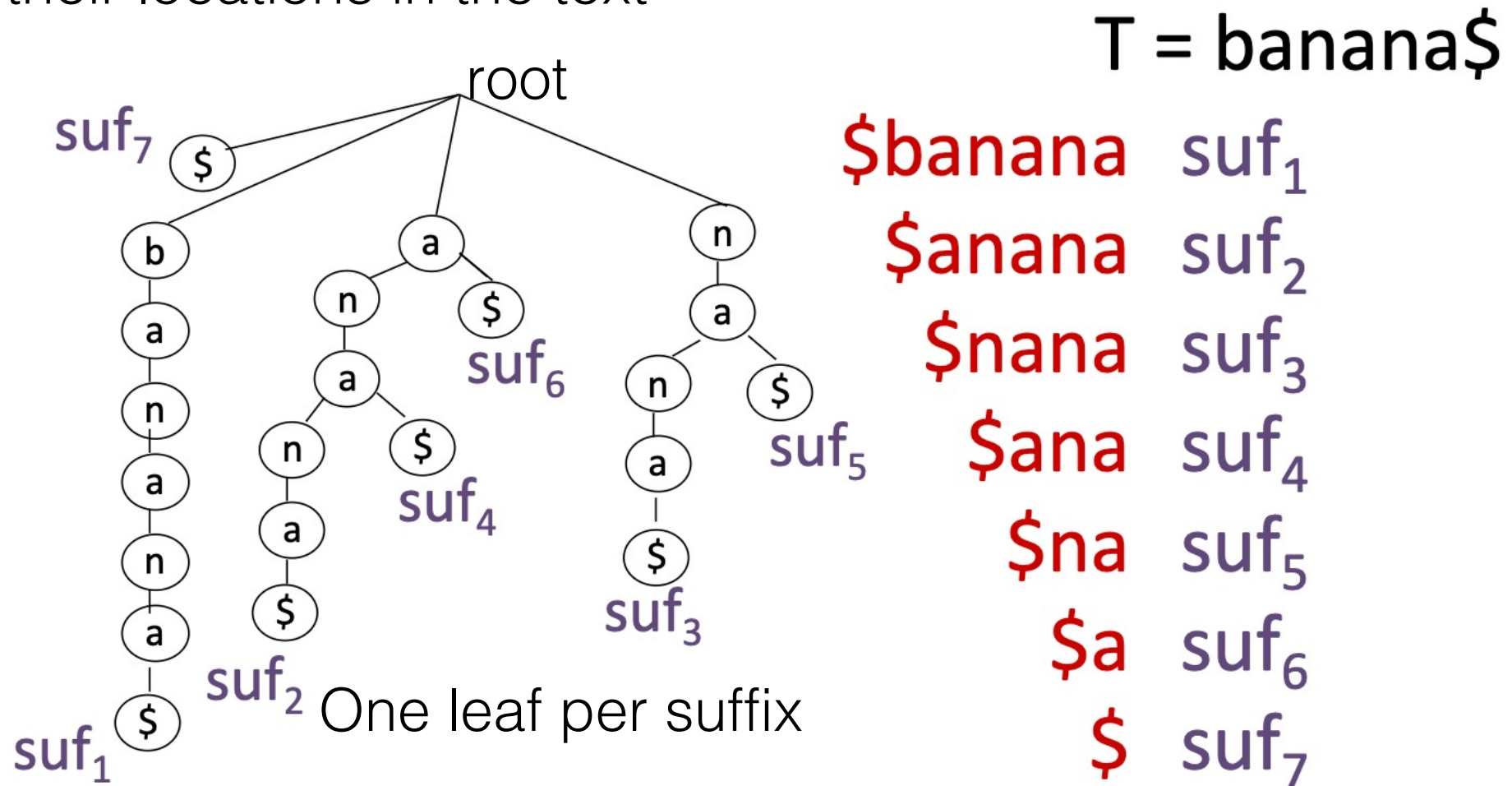


Trapnell &  
Salzberg 2009



# Suffix-Trie

A data structure that contains all suffixes and their locations in the text



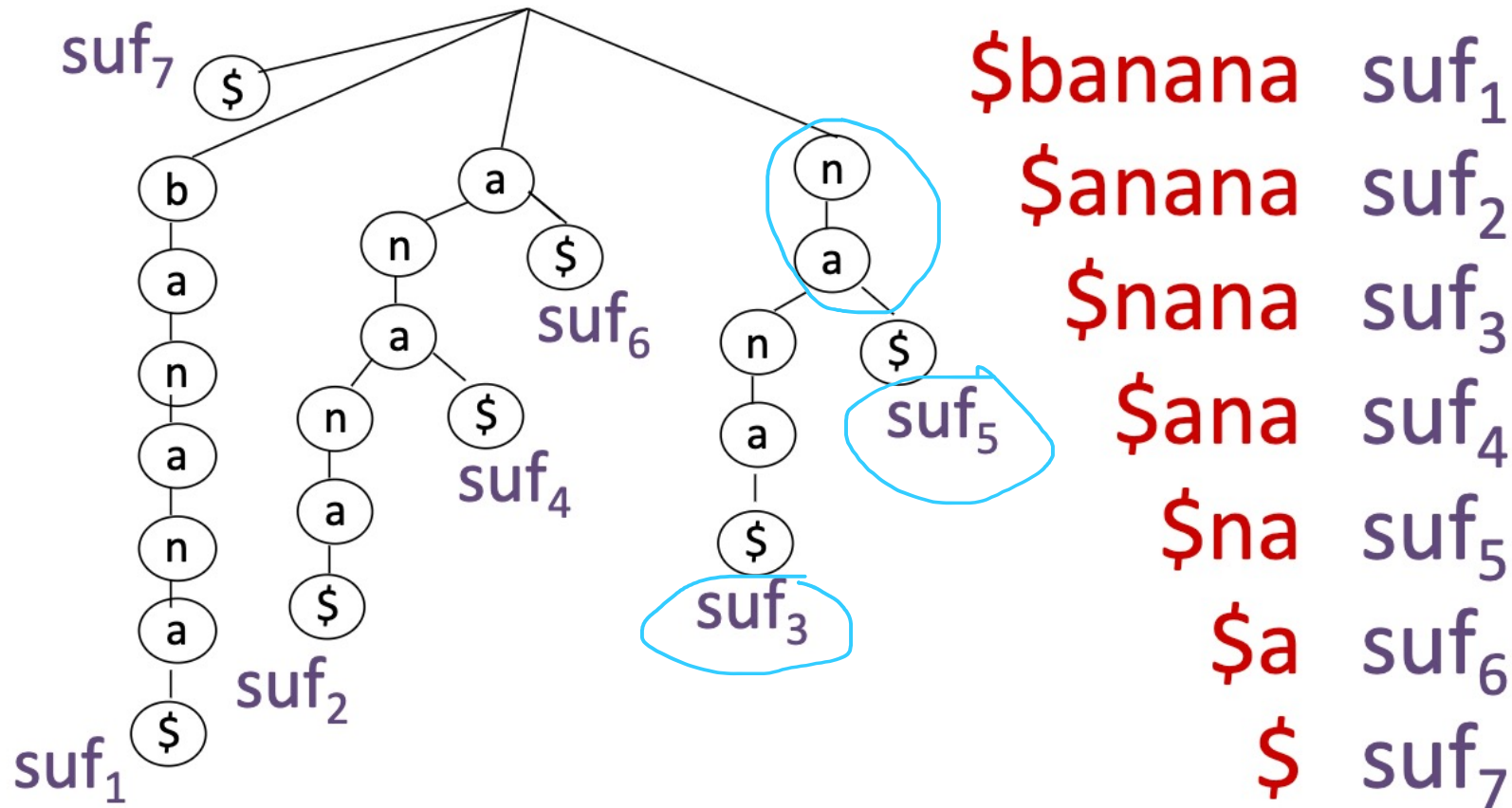
Combining edge labels on the path from the root to the leaf  $i$  spells out the suffix that begins at position  $i$

# Suffix-Trie

Search “na”

A data structure that contains all suffixes and their locations in the text

**T = banana\$**



Suffix tries allow particularly fast implementations of many important string operations (eg. search for substrings quickly)

Try out this website <https://visualgo.net/en/suffixtree>

# Suffix-Prefix Trie

- A family of methods which uses a Trie structure to search a reference sequence (e.g. Bowtie, BWA, SOAP2)
- Trie – data structure which stores the suffixes (i.e. ends of a sequence)
- Key advantage over hashed algorithms:
  - Alignment of multiple copies of an identical sequence in the reference only needs to be done once
  - Use of an FM-Index to store Trie can drastically reduce storage/memory requirements (e.g. Human genome can be stored in 2Gb of RAM)
  - Burrows Wheeler Transform to perform fast lookups

More information about the BWT:

<https://www.youtube.com/watch?v=4n7NPk5lwbl>

# Burrows-Wheeler Transform (BWT)

- Encodes data so that it is easier to compress
- BWT be reversed to recover the original word

$i$	Suffix	Sorted Suffix	SA[ $i$ ]	ISA[ $i$ ]	Sorted Rotations	BWT[ $i$ ]
0	banana\$	\$	6	4	<b>\$banana</b>	a
1	anana\$	a\$	5	3	<b>a\$banan</b>	n
2	nana\$	ana\$	3	6	<b>ana\$ban</b>	n
3	ana\$	anana\$	1	2	<b>anana\$b</b>	b
4	na\$	banana\$	0	5	<b>banana\$</b>	\$
5	a\$	na\$	4	1	<b>na\$bana</b>	a
6	\$	nana\$	2	0	<b>nana\$ba</b>	a

# Burrows-Wheeler Transform (BWT)

- Encodes data so that it is easier to compress
- Can be reversed to recover the original word

$i$	Suffix	Sorted Suffix	SA[ $i$ ]	ISA[ $i$ ]	Sorted Rotations	BWT[ $i$ ]
0	banana\$	\$	6	4	\$banana	a
1	anana\$	a\$	5	3	a\$banan	n
2	nana\$	ana\$	3	6	ana\$ban	n
3	ana\$	anana\$	1	2	anana\$b	b
4	na\$	banana\$	0	5	banana\$	\$
5	a\$	na\$	4	1	na\$bana	a
6	\$	nana\$	2	0	nana\$ba	a

# Burrows-Wheeler Transform (BWT)

- Encodes data so that it is easier to compress
- Can be reversed to recover the original word

$i$	Suffix	Sorted Suffix	SA[ $i$ ]	ISA[ $i$ ]	Sorted Rotations	BWT[ $i$ ]
0	banana\$	\$	6	4	<b>\$banana</b> <sub>1</sub>	a
1	anana\$	<sub>1</sub> a\$	5	3	<b>a\$banan</b> <sub>1</sub>	n
2	nana\$	<sub>2</sub> ana\$	3	6	<b>ana\$ban</b> <sub>2</sub>	n
3	ana\$	<sub>3</sub> anana\$	1	2	<b>anana\$b</b> <sub>1</sub>	b
4	na\$	<sub>1</sub> banana\$	0	5	<b>banana\$</b>	\$
5	a\$	<sub>1</sub> na\$	4	1	<b>na\$bana</b> <sub>2</sub>	a
6	\$	<sub>2</sub> nana\$	2	0	<b>nana\$ba</b> <sub>3</sub>	a



# Burrows-Wheeler Transform (BWT)

- Encodes data so that it is easier to compress
- Can be reversed to recover the original word

$i$	Suffix	Sorted Suffix	SA[ $i$ ]	ISA[ $i$ ]	Sorted Rotations	BWT[ $i$ ]
0	banana\$	\$	6	4	\$banana <sub>1</sub>	a
1	anana\$	<sub>1</sub> a\$	5	3	a\$banan <sub>1</sub>	n
2	nana\$	<sub>2</sub> ana\$	3	6	ana\$ban <sub>2</sub>	n
3	ana\$	<sub>3</sub> anana\$	1	2	anana\$b <sub>1</sub>	b
4	na\$	<sub>1</sub> banana\$	0	5	banana\$	\$
5	a\$	<sub>1</sub> na\$	4	1	na\$bana <sub>2</sub>	a
6	\$	<sub>2</sub> nana\$	2	0	nana\$ba <sub>3</sub>	a

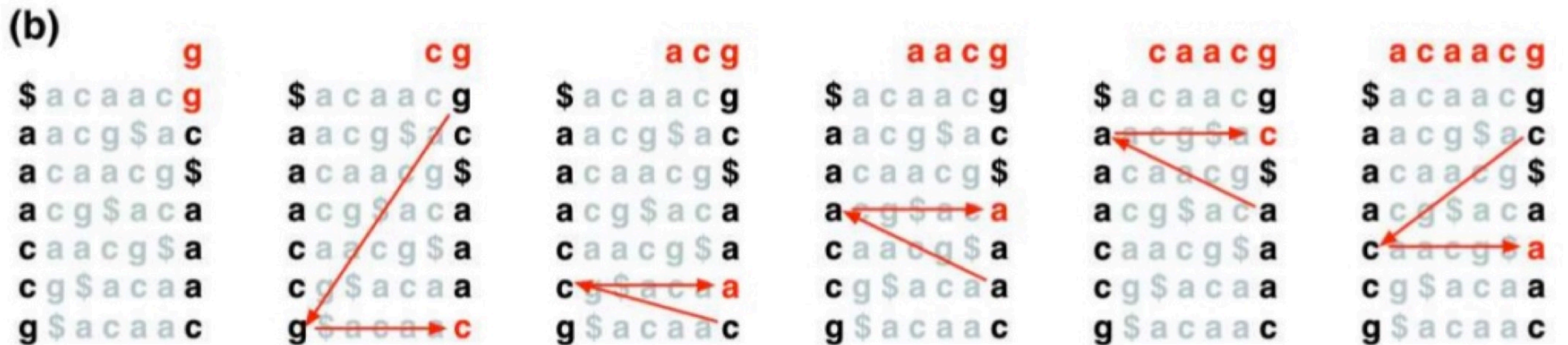
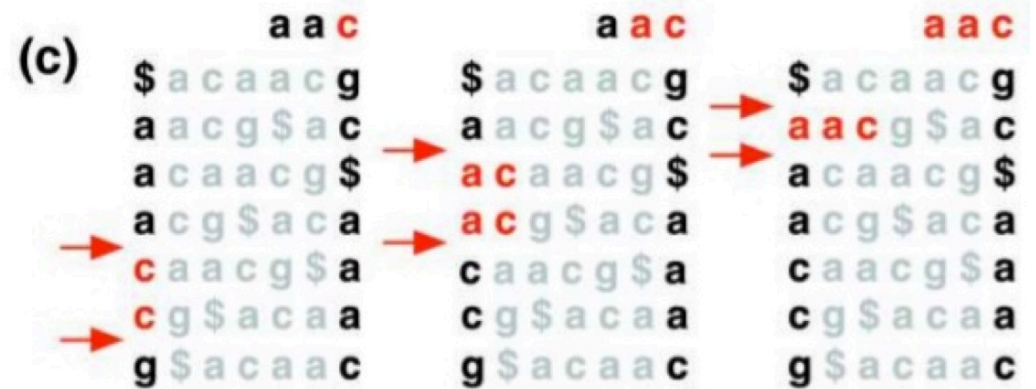
# Burrows-Wheeler Transform (BWT)

- Encodes data so that it is easier to compress
- Can be reversed to recover the original word

$i$	Suffix	Sorted Suffix	SA[ $i$ ]	ISA[ $i$ ]	Sorted Rotations	BWT[ $i$ ]
0	banana\$	\$	6	4	\$banana <sub>1</sub>	a
1	anana\$	<sub>1</sub> a\$	5	3	a\$banan <sub>1</sub>	n
2	nana\$	<sub>2</sub> ana\$	3	6	ana\$ban <sub>2</sub>	n
3	ana\$	<sub>3</sub> anana\$	1	2	anana\$b <sub>1</sub>	b
4	na\$	<sub>1</sub> banana\$	0	5	banana\$	\$
5	a\$	<sub>1</sub> na\$	4	1	na\$bana <sub>2</sub>	a
6	\$	<sub>2</sub> nana\$	2	0	nana\$ba <sub>3</sub>	a



# Burrows-Wheeler Transform



# Suffix-Prefix Trie

- Less sensitive for sequences that are more divergent from the reference.
- Sequencing errors
- Query - Reference differences

# Comparison

Hash referenced spaced seeds (NextGenMap)

- Requires more RAM
- Runs slower
- Simpler to program
- More sensitive

Suffix/Prefix Trie (BWA)

- Requires less RAM
- Runs much faster
- Complicated to program
- Less sensitive

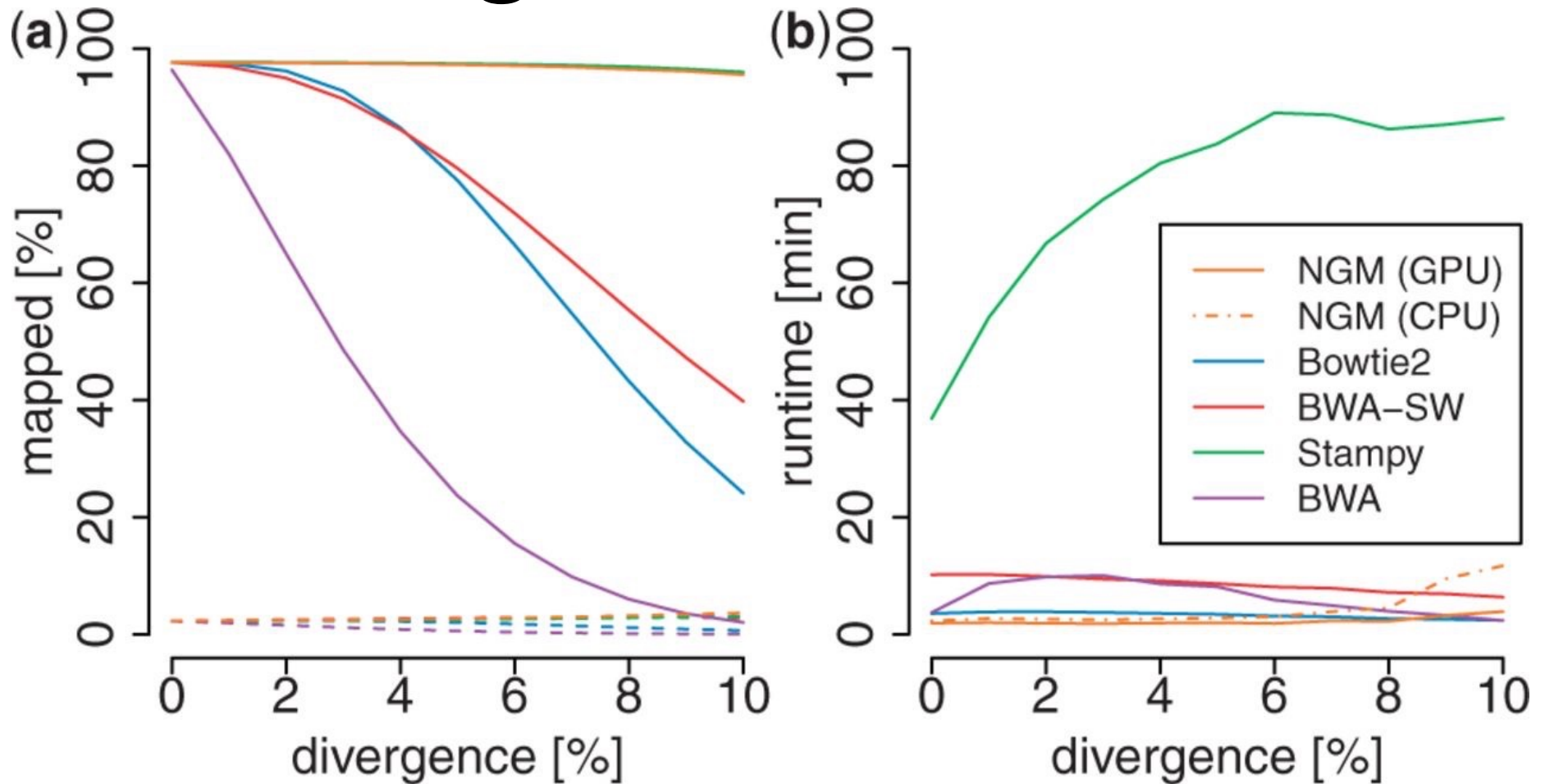
# Popular short read aligners

Program	Algorithm	Speed	Accuracy in for divergent sequences
Bowtie2	Suffix/Prefix	Very fast	Low
BWA	Suffix/Prefix	Fast	Medium
Stampy	Hashing ref	Slow	High
Soap2	Suffix/Prefix	Fast	Low
Novoalign	Hashing ref	Slow	High
NextGenMap	Hashing ref	Med	High

# Think-Pair-Share

- Third generation sequencing can produce very long reads (10-50 Kbp), but are very error prone (~5-10% errors)
- Why would suffix-trie based aligners do poorly with this data?

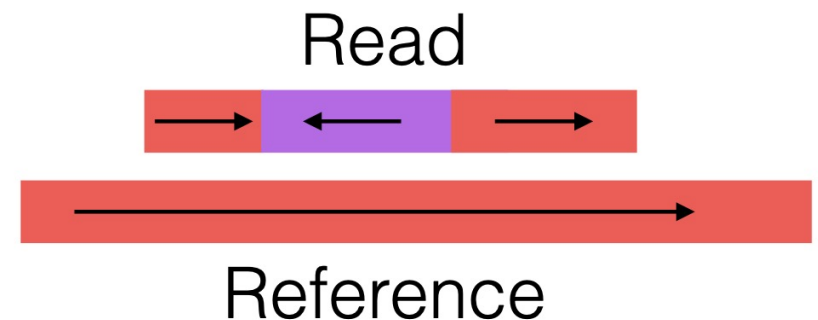
# Alignment stats



\*From NextGenMap paper

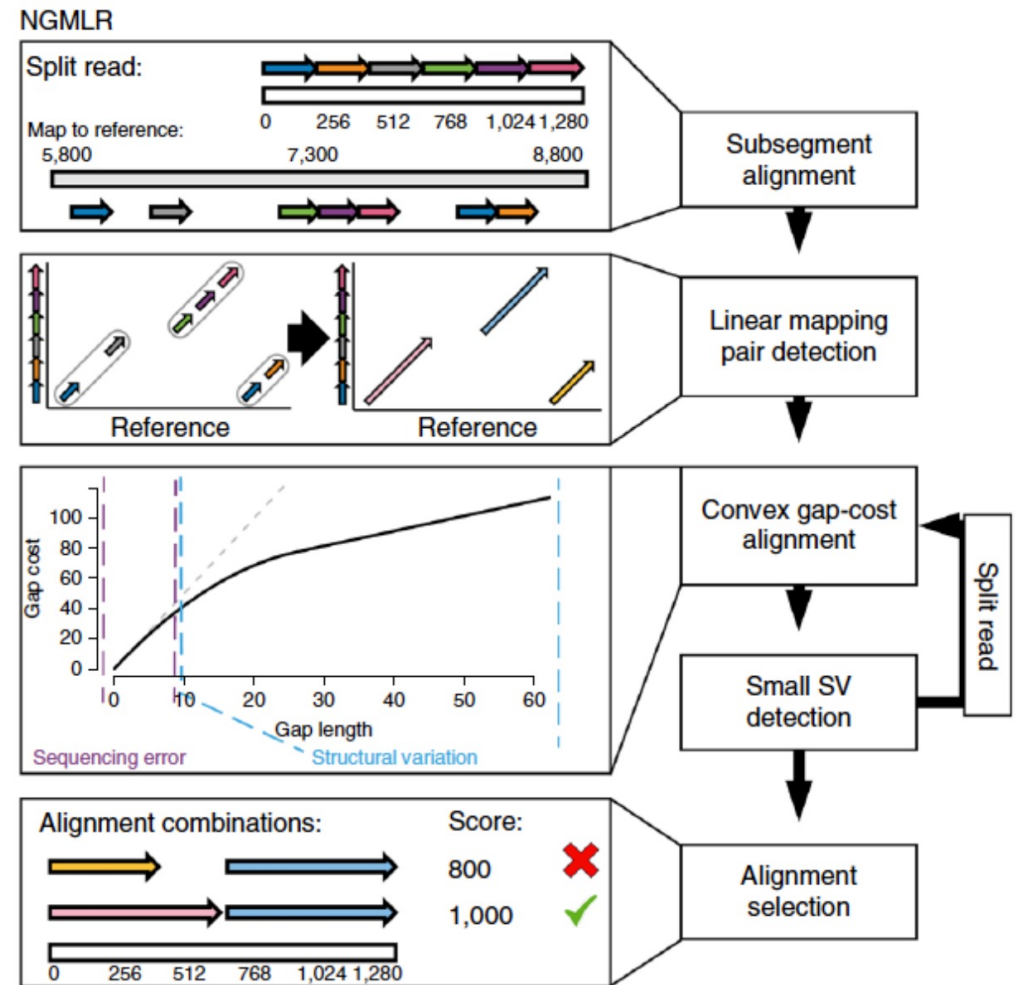
# Long read alignment

- Long reads might contain structural variation that makes it hard to form a linear alignment
- For example, a read containing a large inversion would contain 3 linear alignments
- Most long read technologies have higher error rates



# Long read alignment

1. Find exact matches between read fragment and reference
2. Look for chains of matches
3. Use local alignment of read to best reference region.





# Long read alignment

- Longer reads have more information, but more error.
- Example: **NGMLR** uses k-mers (short strings of a specific length) to pick region and smith-waterman for exact placement.
- Other programs:
  - KART, BWA-MEM, BLASR, minimap2

# Alignment choice

- Speed needed?
- How divergent is sequence from reference? Same species or relative?
- How much variation in your samples?
- Genome size of reference?

# Other considerations

- PCR duplicates
- Multi-mapping reads
- Spliced-read mapping

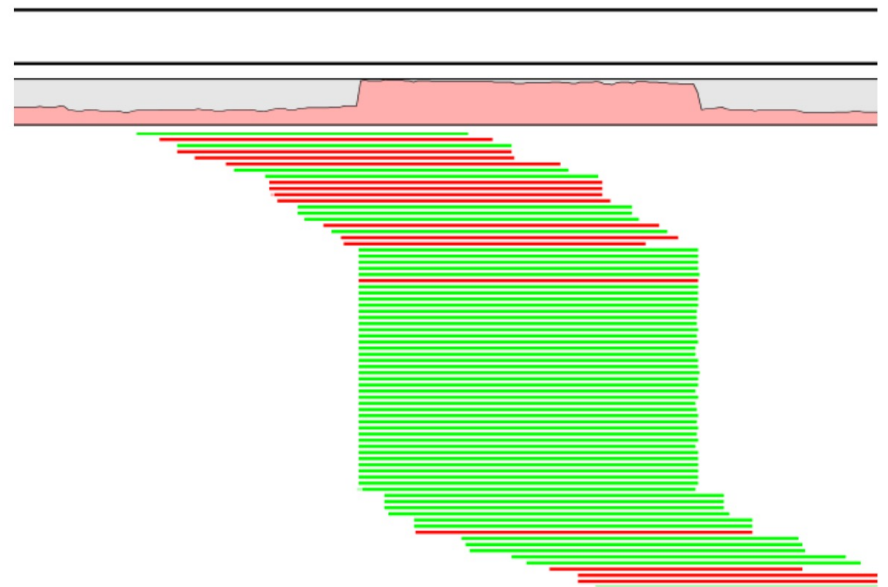
# PCR duplicates

Most library preps have at least one PCR amplification step

PCR can introduce errors and then sequencing multiple copies makes it seem like a real SNP

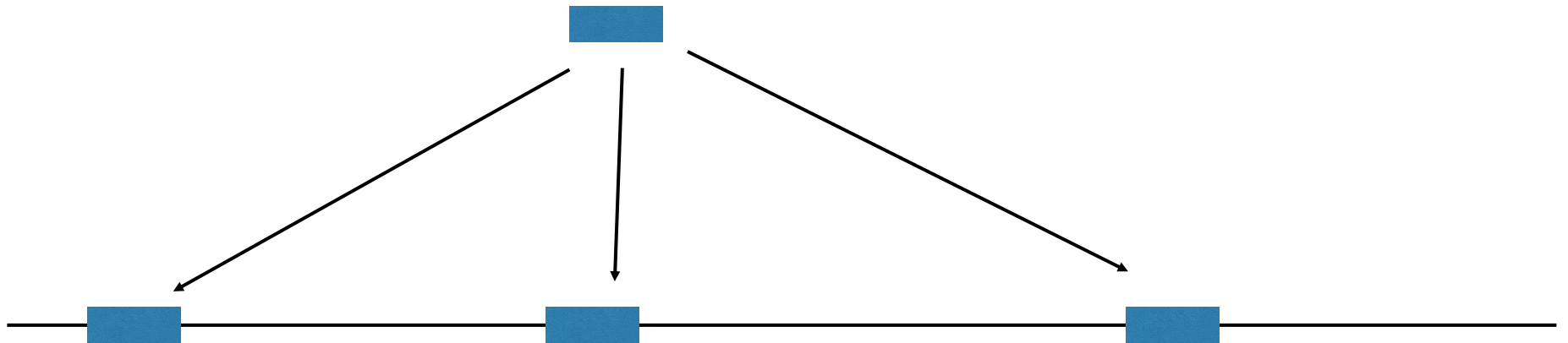
SAMtools and Picard can flag or remove these duplicates based on alignment location

- Samples with same start and stop position are considered duplicates
- Don't flag duplicates for GBS (set start and stop)



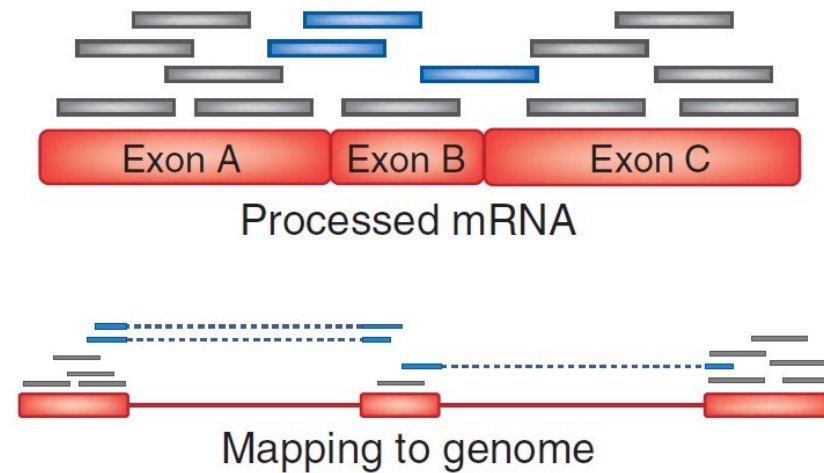
*From Qiagen website*

# Multiple mapping reads



- A single read may occur more than once in a reference genome, due to gene/chromosome duplication or repetitive elements
- Reads may be assigned to one random location
- Affects mapping quality

# Spliced-read mapping



- Need to account for splicing
- Examples: TopHat, SubRead, Star

# SAM (BAM) format

- Sequence Alignment/Map format
  - Universal standard.
  - Generally aligned to reference, but not necessarily
  - Human-readable (SAM) and compressed (BAM) forms

Keeps a track of the reference genome

And individual  
read records



# SAM format

## Header (lines begin with @)

```
@HD VN:1.5 GO:none SO:coordinate
@SQ SN:cp_gi_88656873 LN:151104
@SQ SN:mt_gi_571031384 LN:300945
@SQ SN:rDNA_gi_563582565 LN:9814
@SQ SN:Ha1 LN:175985764
@SQ SN:Ha2 LN:209013747
@SQ SN:Ha3 LN:203472901
@SQ SN:Ha4 LN:216026857
@SQ SN:Ha5 LN:271056985
@SQ SN:Ha6 LN:100519666
@SQ SN:Ha7 LN:109221022
@SQ SN:Ha8 LN:192129815
@SQ SN:Ha9 LN:253478808
@SQ SN:Ha10 LN:327788049
@SQ SN:Ha11 LN:208730832
@SQ SN:Ha12 LN:208068730
@SQ SN:Ha13 LN:239367298
@SQ SN:Ha14 LN:230295834
@SQ SN:Ha15 LN:202246870
@SQ SN:Ha16 LN:226777971
@SQ SN:Ha17 LN:267415242
@SQ SN:Ha0_73Ns LN:359367108
@RG ID:HI.2034.006.Index_18.W70_NHK_2013_5 LB:Anomalus PL:ILLUMINA SM:HI.2034.006.Index_18.W70_NHK_2013_5 PU:Anomalus
@PG ID:ngm PN:ngm CL:" --affine 0 --argos_min_score 0 --bam 1 --block_multiplier 2 --bs_cutoff 6 --bs_mapping 0 --cpu_threads 11 --dualstrand 1
@PG ID:ngm.1 PN:ngm CL:" --affine 0 --argos_min_score 0 --bam 1 --block_multiplier 2 --bs_cutoff 6 --bs_mapping 0 --cpu_threads 11 --
@PG ID:ngm.2 PN:ngm CL:" --affine 0 --argos_min_score 0 --bam 1 --block_multiplier 2 --bs_cutoff 6 --bs_mapping 0 --cpu_threads 11 --
```

Sort order

Reference sequence name (SN) and length (LN) e.g. Chromosome Ha7, which is 109,221,022bp long

Read group information (@RG)

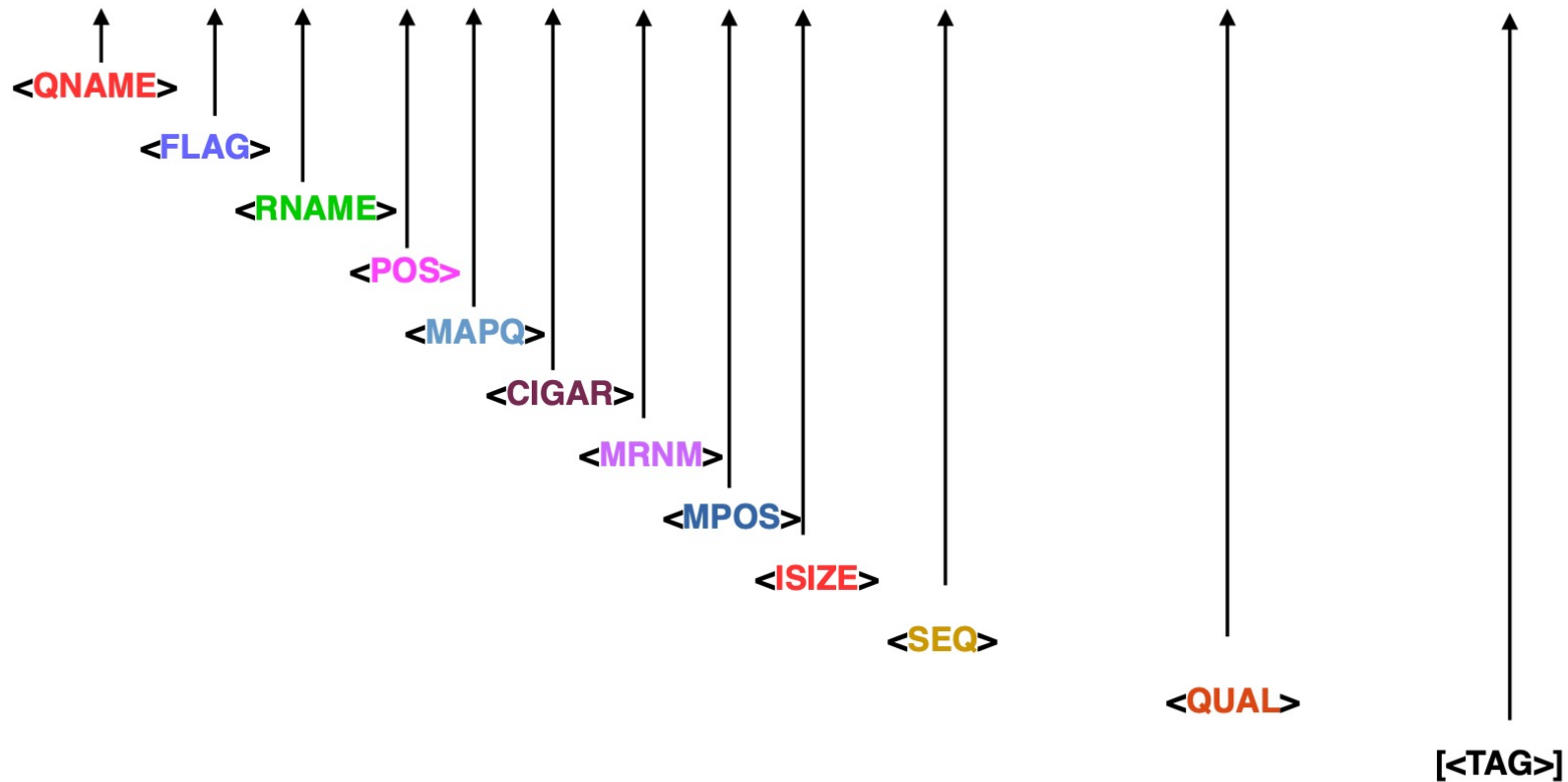
Program (@PG) information - what you used to map the reads



# SAM format

## Alignment lines

SRR035022 163 chr16 59999 37 22D54M = 60102 179 CCAACCCAAC... >AAA=>?AA... XT:A:M XN:i:2 SM:i:37



# SAM format

## Alignment lines

**SRR035022** **163** **chr16** **59999** **37** **22D54M** **=** **60102** **179** **CCAACCCAAC...** **>AAA=>?AA...** **XT:A:M** **XN:i:2** **SM:i:37**

**<QNAME>** Query name - i.e. the name of the read

**<FLAG>** A combination of bitwise flags that indicate properties of the alignment (complement, strand etc.)

**<RNAME>** Reference sequence name

**<POS>** Position in the reference (1-based)

**<MAPQ>** Mapping quality

**<CIGAR>** Concise Idiosyncratic Gapped Alignment Report (CIGAR) string - tells you about gaps in the alignment

**<MRNM>** Read-mate reference sequence

**<MPOS>** Read-mate position in the reference

**<ISIZE>** Insert size

**<SEQ>** The segment's sequence

**<QUAL>** An ASCII sequence containing quality information for each base in the sequence

**[<TAG>]** These are optional tags that get added and contain user specified data (For example, SM is the mapping quality of this sequence only - ignoring the read mate)

# Mapping Quality

- $\text{MapQ} = Q_s = -10 \log_{10}(P)$
- $P$  = probability that this mapping is NOT the correct one
- $\text{MapQ} = 0$  = equally likely to map somewhere else
- Different programs use different formulas for  $P$
- A value of 255 indicates that the information is not available

# Background reading

- Trapnell, C., & Salzberg, S. L. (2009). How to map billions of short reads onto genomes. *Nature biotechnology*, 27(5), 455-457.
- Reinert, K., Langmead, B., Weese, D., & Evers, D. J. (2015). Alignment of next-generation sequencing reads. *Annual review of genomics and human genetics*, 16, 133-151.

All available on the GitHub

# Burrows-Wheeler Algorithm

- Encodes data so that it is easier to compress
- Can be reversed to recover the original word

$i$	Suffix	Sorted Suffix	SA[ $i$ ]	ISA[ $i$ ]	Sorted Rotations	BWT[ $i$ ]
0	banana\$	\$	6	4	\$banana <sub>1</sub>	a
1	anana\$	a\$	5	3	a\$banan <sub>1</sub>	n
2	nana\$	ana\$	3	6	ana\$ban <sub>2</sub>	n
3	ana\$	anana\$	1	2	anana\$b <sub>1</sub>	b
4	na\$	banana\$	0	5	banana\$	\$
5	a\$	na\$	4	1	na\$bana <sub>2</sub>	a
6	\$	nana\$	2	0	nana\$ba <sub>3</sub>	a