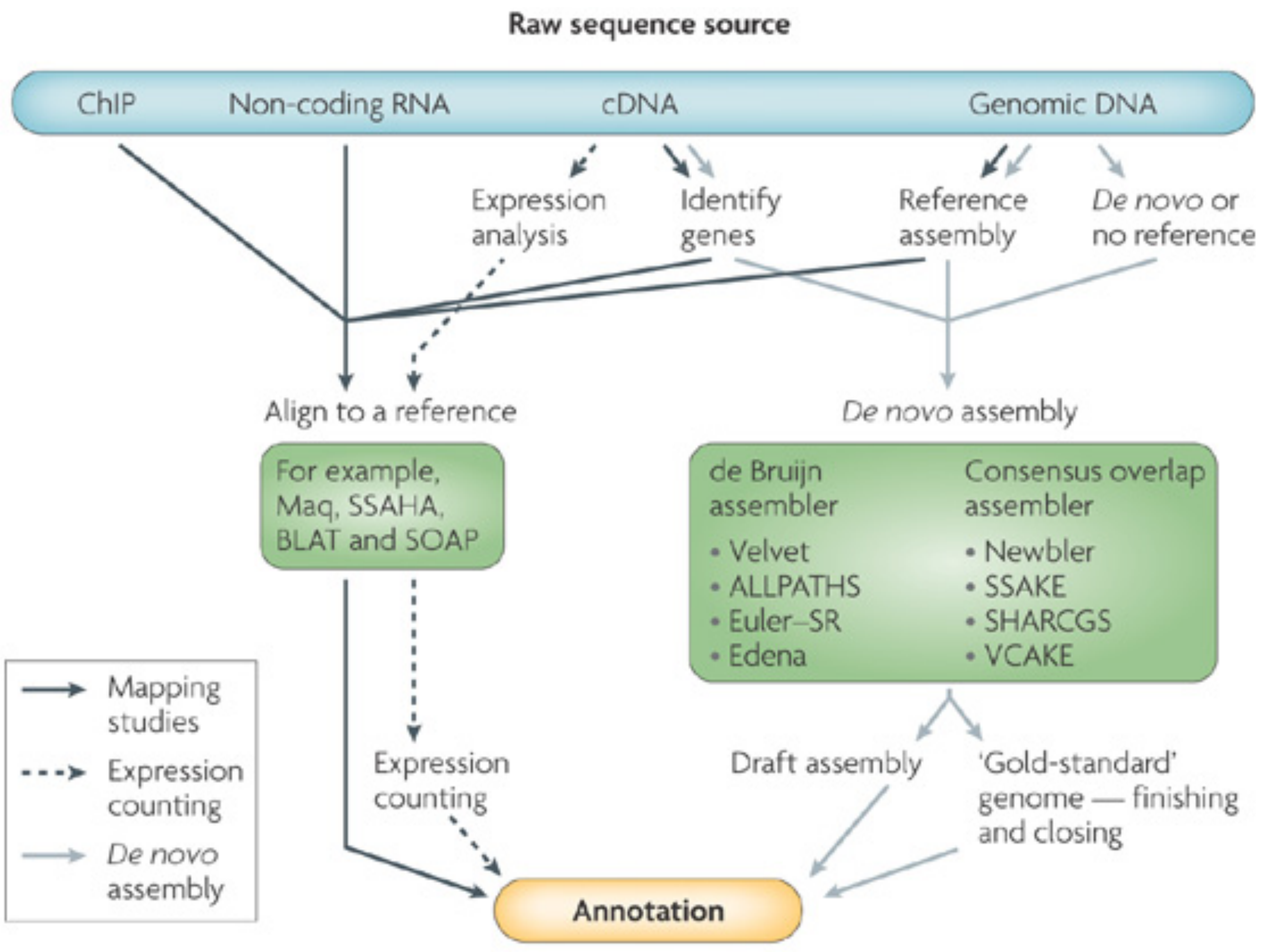
Workshop on Genomics 2015 Sequence Alignment: An brief introduction

Konrad Paszkiewicz
University of Exeter, UK
k.h.paszkiewicz@exeter.ac.uk

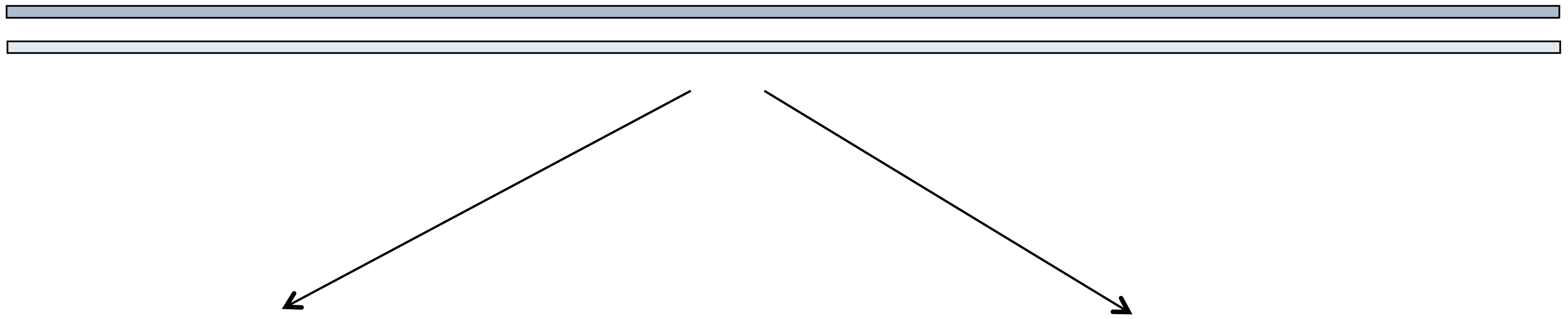
Contents

- **Alignment algorithms for short-reads**
  - Background – Blast (why can’t we use it?)
  - Adapting hashed seed-extend algorithms to work with shorter reads
  - Suffix/Prefix Tries
  - Indels
  - Other alignment considerations
  - Typical alignment pipeline
  - Haplotype methods of variant calling



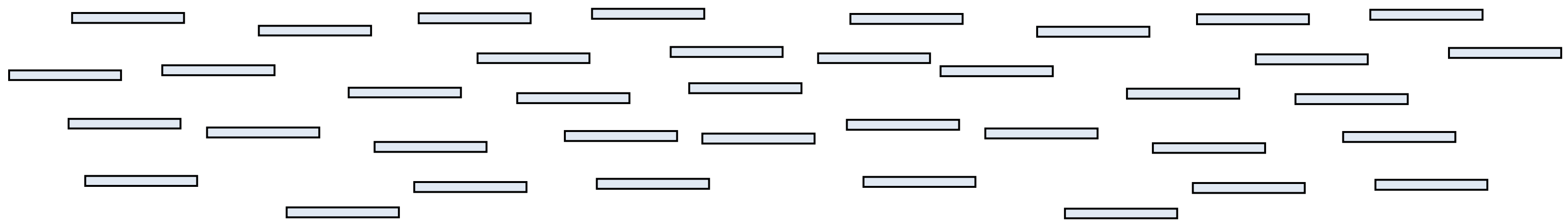
Nature Reviews | Microbiology

Alignment of reads to a reference



..ACTGGGTCATCGTACGATCGATCGATCGATCGATCGGCTAGCTAGCTA.. Reference

..ACTGGGTCATCGTACGATCGATAGATCGATCGATCGCTAGCTAGCTA.. Sample



Why is short read alignment hard?

The shorter a read, the less likely it is to have a unique match to a reference sequence

Need over 7kb reads to uniquely place most reads from bacteria

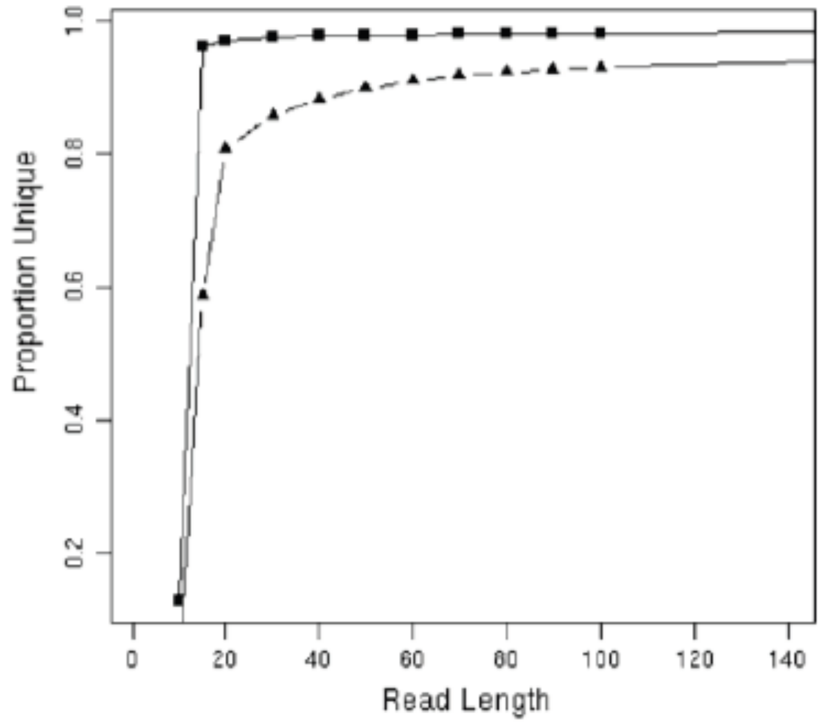


Fig. 1 The proportion of unique sequence in the *Streptococcus suis* (squares) and *Mus musculus* (triangles) genomes for varying read lengths. This graph indicates that read length has a critical affect on the ability to place reads uniquely to the genome

Why do we generate short reads?

- Sanger reads lengths ~ 800-2000bp

- Generally we define short reads as anything below this
  - Illumina (50bp – 300bp)
  - SoLID (80bp max)
  - Ion Torrent (200-400bp max...)
  - Roche 454 – 400-800bp

- Even with these platforms it is cheaper to produce short reads (e.g. 50bp) rather than 100 or 200bp reads

- Diminishing returns:
  - For some applications 50bp is more than sufficient
    - Small-RNA
    - ChIP-Seq
    - Differential gene expression
    - Digital Gene Expression profiling

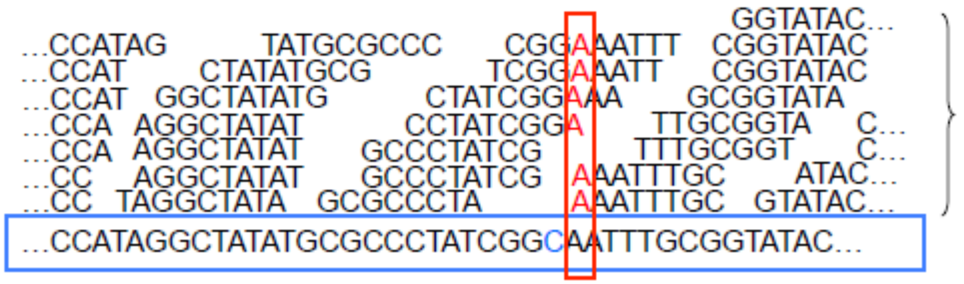
Short read alignment applications

Genotyping:

Methylation

SNPs

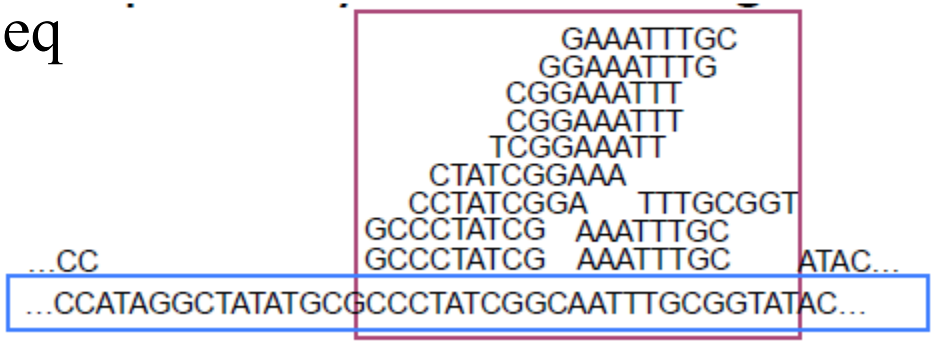
Indels


GGTATAC...
...CCATAG TATGCGCCC CGGAAATTT CGGTATAC
...CCAT CTATATGCG TCGGAAATT CGGTATAC
...CCAT GGCTATATG CTATCGGAAA GCGGTATA
...CCA AGGCTATAT CCTATCGGA TTGCGGTA C...
...CCA AGGCTATAT GCCCTATCG TTTGCGGT C...
...CC AGGCTATAT GCCCTATCG AAATTTGC ATAC...
...CC TAGGCTATA GCGCCCTA AAATTTGC GTATAC...
...CCATAGGCTATATGCGCCCTATCGGCAATTTGCGGTATAC...

Classify and measure peaks:

ChIP-Seq

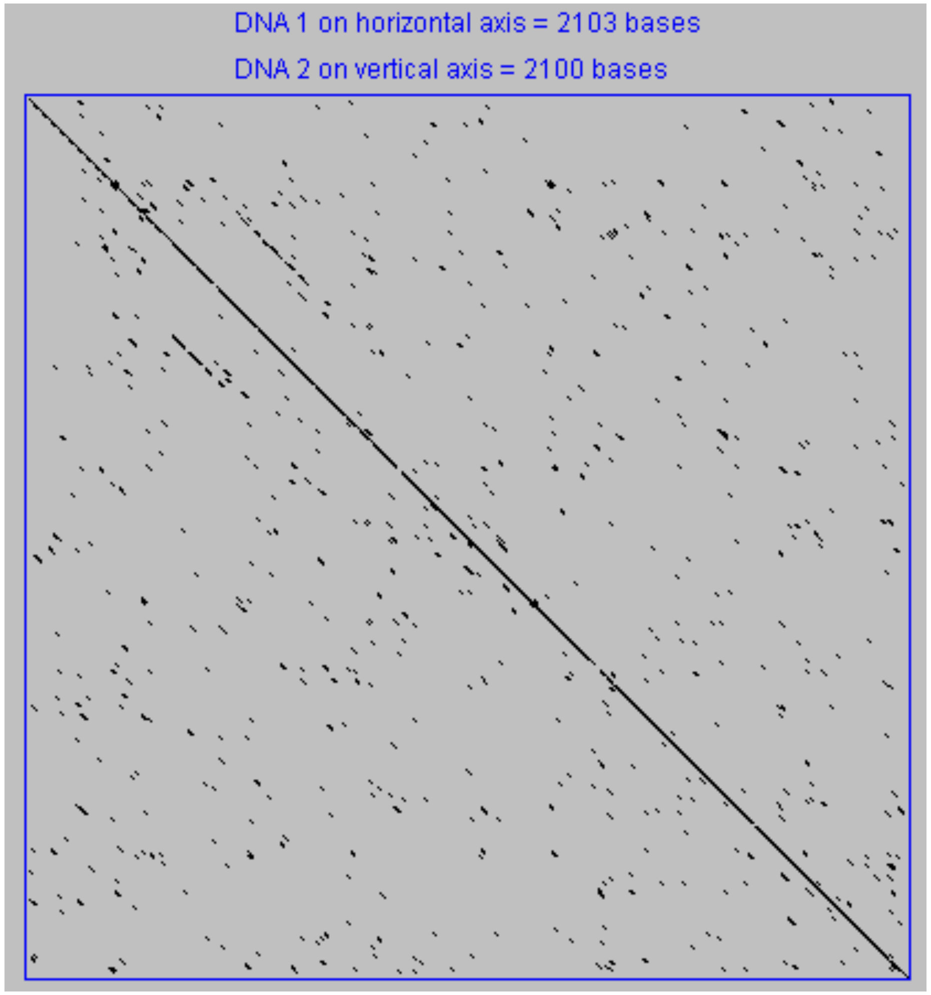
RNA-Seq



Contents

- **Alignment algorithms for short-reads**
  - **Background – Blast (why can't we use it?)**
    - Global alignment
    - Local alignment
  - Adapting hashed seed-extend algorithms to work with shorter reads
  - Indel detection
  - Suffix/Prefix Tries
  - Other alignment considerations
  - Typical alignment pipeline
  - Haplotype methods of variant calling

Dot Matrix Method
- Aligning by eye



Sequence Alignment

```
ATCGATA-CG
ATGGATTACG
```

3 possibilities

Match	Mismatch	Indel
...A...	...C...	...-...
|		
...A...	...G...	...T...

A very simple alignment scoring system

Points for a matching letter:	1
Points for a non-matching letter:	0
Points for inserting a gap:	0

Global Pair-wise Alignment

ATCGATACG, ATGGATTACG

```
ATCGAT-ACG
|| ||| |||
ATGGATTACG
```

Matches:	+1	+1		+1	+1	+1		+1	+1	+1	= +8
Mismatches:			0								= 0
Gaps:							0				= 0

Total score = +8

But, what does this score mean??
Could we get a better alignment?

How to choose the best alignment?

- Sequence 1: `ACTGAGC`
- Sequence 2: `ATGATGC`

- Some possible alignments:

```
ACTGAGC-- ACTGA-GC A----CTGAGC
A-TGA-TGC A-TGATGC ATGAT----GC
```

Global alignment – Needleman-Wunsch

A **global** alignment covers the **entire lengths of the sequences** involved

The Needleman-Wunsch algorithm finds the best global alignment between 2 sequences across their whole length

Step 1: Initialise

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

Fill in far-right column and bottom row with:

0 for a mis-match

1 for a match

Step 2:

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

For each box, find the highest number out of the blue boxes

Step 3:

	A	C	T	G	A	G	C
A							0
T							0
G							0
A							0
T							0
G						1+1=2	0
C	0	1	0	0	0	0	1

If there is a match in the yellow box as, take the highest value from the blue boxes and add 1 to it
G matches G in the yellow box, so add 1 to the 1 in the blue box

Step 2:

	A	C	T	G	A	G	C
A							0
T							0
G							0
A							0
T							0
G					0+1=1	2	0
C	0	1	0	0	0	0	1

A does not match G. So add zero to the 1 in the blue box.

Step 2:

	A	C	T	G	A	G	C
A							0
T							0
G							0
A							0
T							0
G				1+1=2	1	2	0
C	0	1	0	0	0	0	1

If there is a match as here, take the highest value and add 1 to it

G matches G so add 1 to 1 in the blue boxes

Step 2:

	A	C	T	G	A	G	C
A							0
T							0
G							0
A							0
T							0
G			0+1=1	2	1	2	0
C	0	1	0	0	0	0	1

If there is a match as here, take the highest value and add 1 to it

T does not match G. So add zero.

Step 2:

	A	C	T	G	A	G	C
A							0
T							0
G							0
A							0
T						0+1=1	0
G	1	1	1	2	1	2	0
C	0	1	0	0	0	0	1

Highest out of the blue boxes is 1

.

Step 2:

	A	C	T	G	A	G	C
A							0
T							0
G							0
A							0
T					2+0=2	1	0
G	1	1	1	2	1	2	0
C	0	1	0	0	0	0	1

Highest out of the blue boxes is 2

A does not match T

Step 2:

	A	C	T	G	A	G	C
A							0
T							0
G							0
A							0
T				2+0=2	2	1	0
G	1	1	1	2	1	2	0
C	0	1	0	0	0	0	1

Highest out of the blue boxes is 2

G does not match T

Step 2:

	A	C	T	G	A	G	C
A							0
T							0
G							0
A							0
T			3	2	2	1	0
G	1	1	1	2	1	2	0
C	0	1	0	0	0	0	1

Highest out of the blue boxes is 2

T **does** match T

Step 2:

	A	C	T	G	A	G	C
A							0
T							0
G							0
A							0
T		2+0=2	3	2	2	1	0
G	1	1	1	2	1	2	0
C	0	1	0	0	0	0	1

Highest out of the blue boxes is 2

C does not match T

Step 2:

	A	C	T	G	A	G	C
A							0
T							0
G							0
A							0
T	2	2	3	2	2	1	0
G	1	1	1	2	1	2	0
C	0	1	0	0	0	0	1

Do the same for all remaining rows

.

Step 2:

	A	C	T	G	A	G	C
A							0
T							0
G							0
A						1+0=1	0
T	2	2	3	2	2	1	0
G	1	1	1	2	1	2	0
C	0	1	0	0	0	0	1

Do the same for all remaining rows

.

Step 2:

	A	C	T	G	A	G	C
A							0
T							0
G							0
A					2+1=3	1	0
T	2	2	3	2	2	1	0
G	1	1	1	2	1	2	0
C	0	1	0	0	0	0	1

Do the same for all remaining rows

.

Step 2:

	A	C	T	G	A	G	C
A							0
T							0
G							0
A				2+0=2	3	1	0
T	2	2	3	2	2	1	0
G	1	1	1	2	1	2	0
C	0	1	0	0	0	0	1

Do the same for all remaining rows

.

Step 2:

	A	C	T	G	A	G	C
A							0
T							0
G							0
A			2+0=2	2	3	1	0
T	2	2	3	2	2	1	0
G	1	1	1	2	1	2	0
C	0	1	0	0	0	0	1

Do the same for all remaining rows

.

Step 2:

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

Do the same for all remaining rows

.

Step 3: Backtracking

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

Follow largest numbers starting from top-left going down and to the right

.

Step 3: Backtracking

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

Follow largest numbers starting from top-left going down and to the right

.

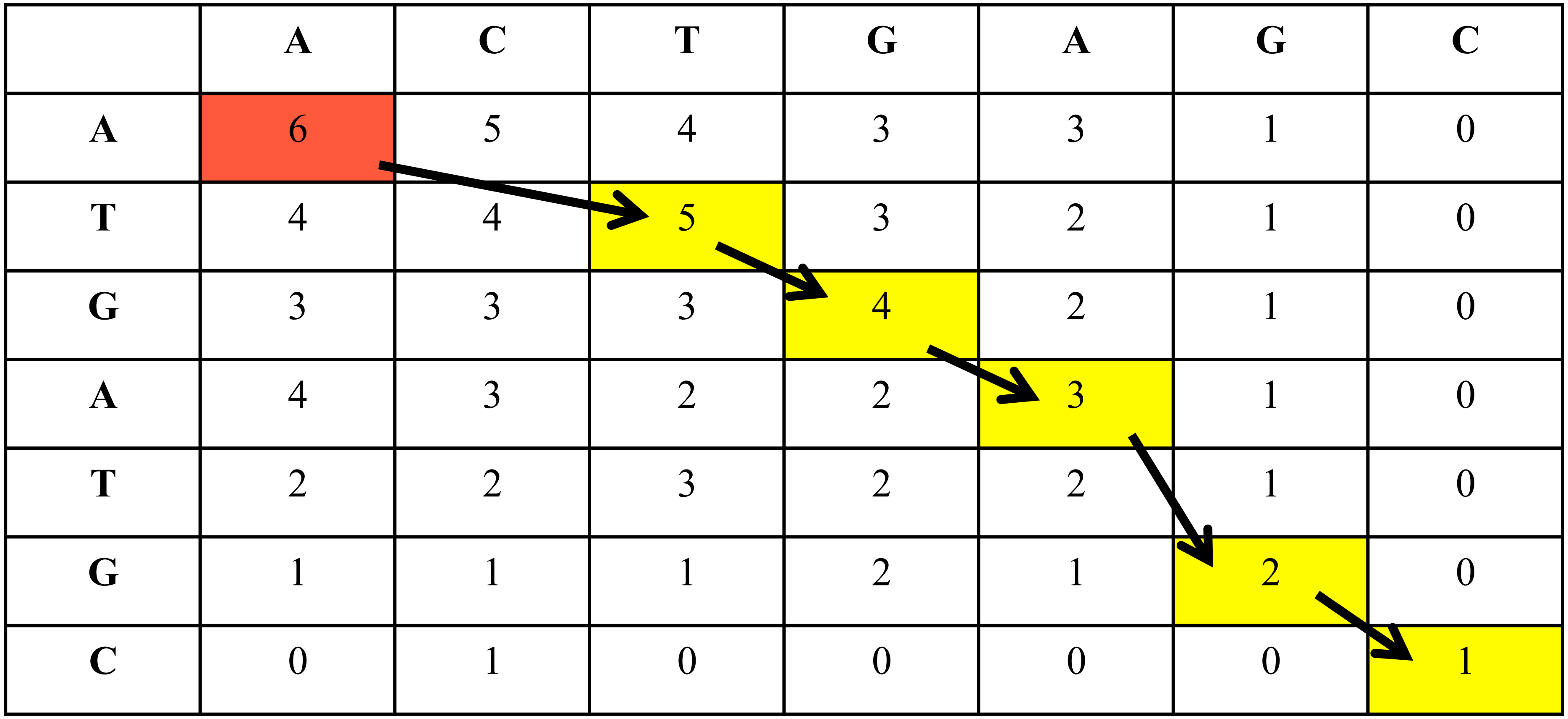
Step 3: Backtracking

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

Follow **largest** numbers starting from **top-left,**
going down and to the **right**

.

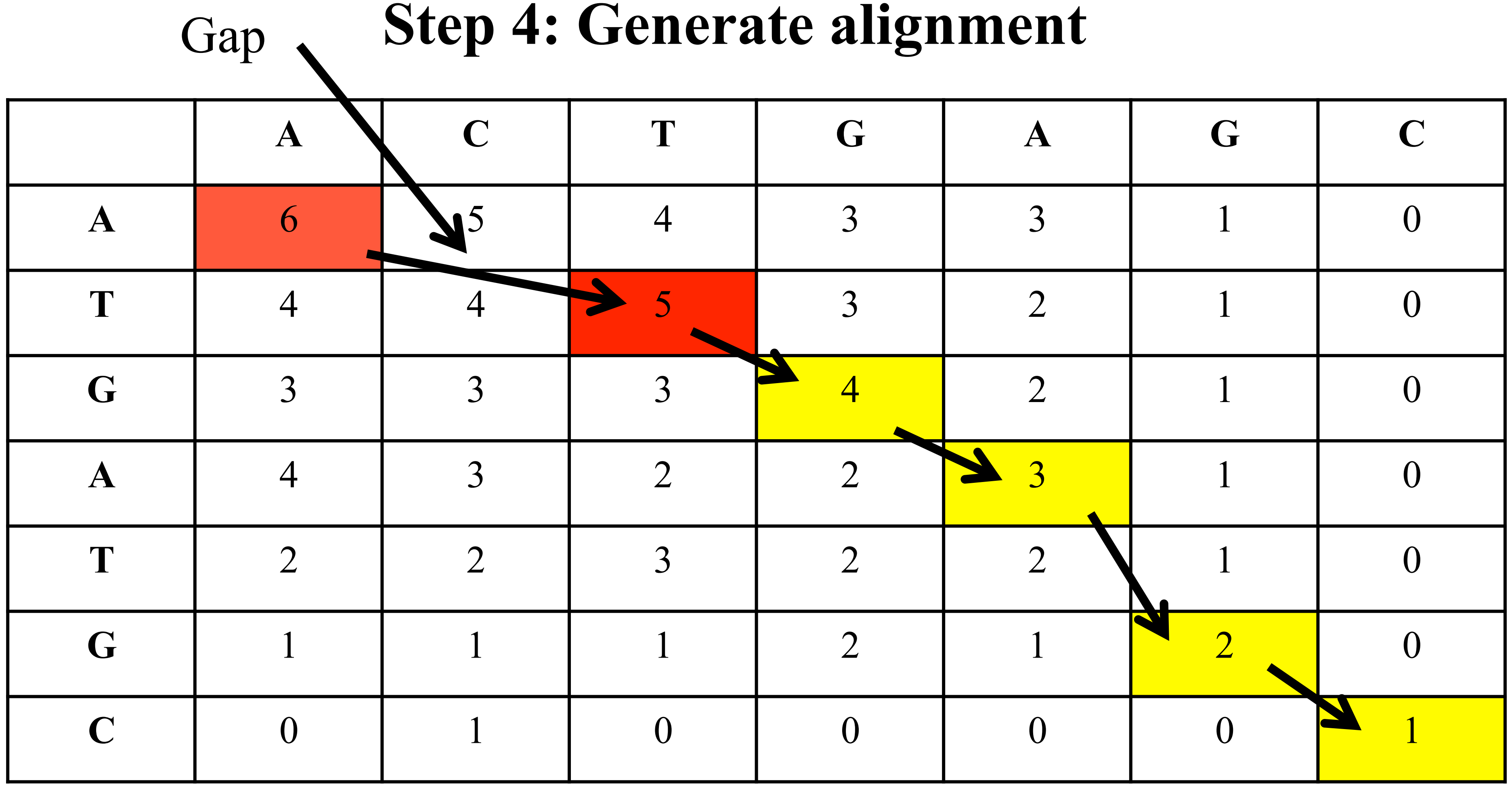
Step 4: Generate alignment



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

```
Horizontal seq      A
Vertical seq        A
```

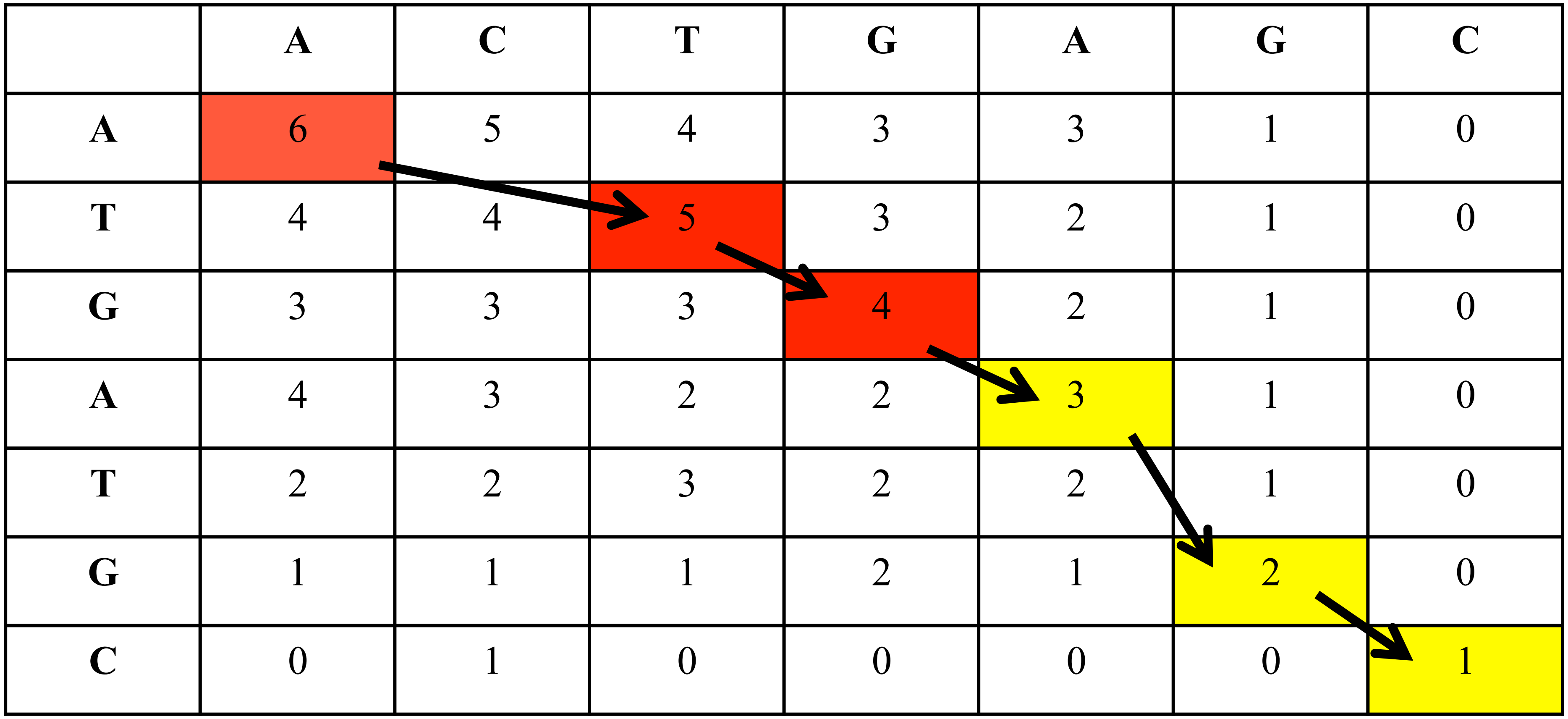
Step 4: Generate alignment



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

```
Horizontal seq      ACT
Vertical seq        A-T
```

Step 4: Generate alignment



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

```
Horizontal seq      ACTG
Vertical seq        A-TG
```

Step 4: Generate alignment

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

```
Horizontal seq      ACTGA
Vertical seq        A-TGA
```

Step 4: Generate alignment

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

```
Horizontal seq      ACTGA-
Vertical seq        A-TGAG
```

Step 4: Generate alignment

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

```
Horizontal seq      ACTGA-C
Vertical seq        A-TGAGC
```

Optimal global alignment

```
ACTGA-C
| ||| |
A-TGAGC
```

Local alignment

A **global** alignment is often not appropriate as only parts of sequences may be conserved

A **local** alignment only covers **parts of the sequences**

The **Smith-Waterman** algorithm finds the **best local alignment** between 2 sequences

Global alignment

```
  Q K E S G P S S S Y C
  |   | | |           |
V Q Q E S G L V R T T C
```

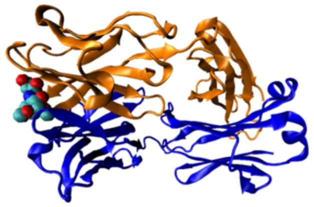
Local alignment

```
E S G
| | |
E S G
```

Local alignment

A **local alignment** of 2 sequences is an alignment between **parts** of the 2 sequences

E.g. Two proteins may be very similar in a functional site, but be very dissimilar outside that region

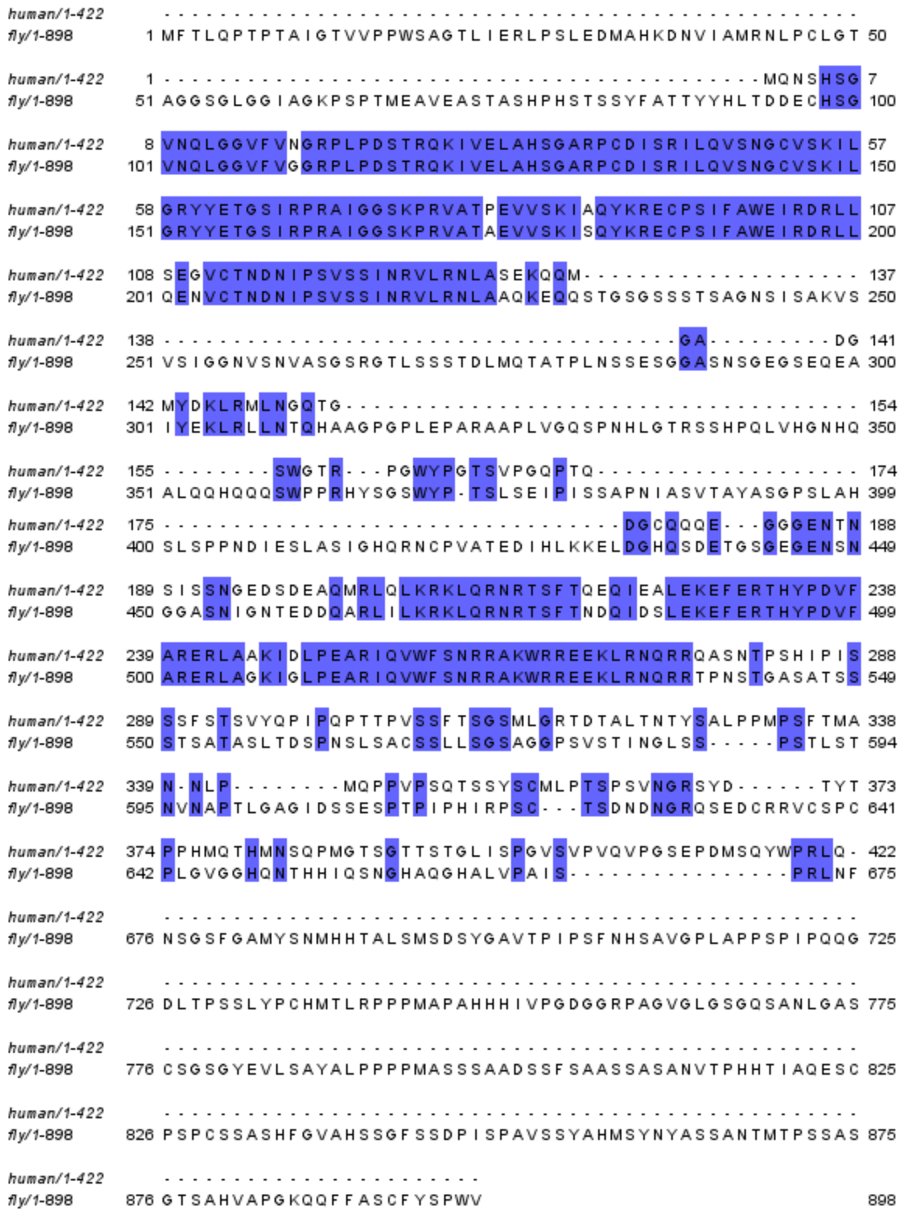


A global alignment of such sequences would have:

(i) lots of matches in the region of high sequence similarity

(ii) lots of mismatches & gaps (insertions/deletions) outside the region of similarity

It makes sense to find the **best local alignment** instead



```
human/1-422      --------------------------------------------------
fly/1-898      1 MFTLQPTPTAIGTVVPPWSAGTLIERLPSLEDMAHKDNVIAMRNLPCLGT 50

human/1-422    1 -------------------------------------------MQNSHSG 7
fly/1-898     51 AGGSGLGGIAGKPSPTMEAVEASTASHPHSTSSYFATTYYHLTDDECHSG 100

human/1-422    8 VNQLGGVFVNGRPLPDSTRQKIVELAHSGARPCDISRILQVSNGCVSKIL 57
fly/1-898    101 VNQLGGVFVGGRPLPDSTRQKIVELAHSGARPCDISRILQVSNGCVSKIL 150

human/1-422   58 GRYYETGSIRPRAIGGSKPRVATPEVVSKIAQYKRECPSIFAWEIRDRLL 107
fly/1-898    151 GRYYETGSIRPRAIGGSKPRVATAEVVSKISQYKRECPSIFAWEIRDRLL 200

human/1-422  108 SEGVCTNDNIPSVSSINRVLRNLASEKQQM-------------------- 137
fly/1-898    201 QENVCTNDNIPSVSSINRVLRNLAAQKEQQSTGSGSSSTSAGNSISAKVS 250

human/1-422  138 -------------------------------------GA---------DG 141
fly/1-898    251 VSIGGNVSNVASGSRGTLSSSTDLMQTATPLNSSESGGASNSGEGSEQEA 300

human/1-422  142 MYDKLRMLNGQTG------------------------------------- 154
fly/1-898    301 IYEKLRLLNTQHAAGPGPLEPARAAPLVGQSPNHLGTRSSHPQLVHGNHQ 350

human/1-422  155 --------SWGTR---PGWYPGTSVPGQPTQ------------------- 174
fly/1-898    351 ALQQHQQQSWPPRHYSGSWYP-TSLSEIPISSAPNIASVTAYASGPSLAH 399
human/1-422  175 ---------------------------------DGCQQQE---GGGENTN 188
fly/1-898    400 SLSPPNDIESLASIGHQRNCPVATEDIHLKKELDGHQSDETGSGEGENSN 449

human/1-422  189 SISSNGEDSDEAQMRLQLKRKLQRNRTSFTQEQIEALEKEFERTHYPDVF 238
fly/1-898    450 GGASNIGNTEDDQARLILKRKLQRNRTSFTNDQIDSLEKEFERTHYPDVF 499

human/1-422  239 ARERLAAKIDLPEARIQVWFSNRRAKWRREEKLRNQRRQASNTPSHIPIS 288
fly/1-898    500 ARERLAGKIGLPEARIQVWFSNRRAKWRREEKLRNQRRTPNSTGASATSS 549

human/1-422  289 SSFSTSVYQPIPQPTTPVSSFTSGSMLGRTDTALTNTYSALPPMPSFTMA 338
fly/1-898    550 STSATASLTDSPNSLSACSSLLSGSAGGPSVSTINGLSS-----PSTLST 594

human/1-422  339 N-NLP--------MQPPVPSQTSSYSCMLPTSPSVNGRSYD------TYT 373
fly/1-898    595 NVNAPTLGAGIDSSESPTPIPHIRPSC---TSDNDNGRQSEDCRRVCSPC 641

human/1-422  374 PPHMQTHMNSQPMGTSGTTSTGLISPGVSVPVQVPGSEPDMSQYWPRLQ- 422
fly/1-898    642 PLGVGGHQNTHHIQSNGHAQGHALVPAIS----------------PRLNF 675

human/1-422      --------------------------------------------------
fly/1-898    676 NSGSFGAMYSNMHHTALSMSDSYGAVTPIPSFNHSAVGPLAPPSPIPQQG 725

human/1-422      --------------------------------------------------
fly/1-898    726 DLTPSSLYPCHMTLRPPPMAPAHHHIVPGDGGRPAGVGLGSGQSANLGAS 775

human/1-422      --------------------------------------------------
fly/1-898    776 CSGSGYEVLSAYALPPPPMASSSAADSSFSAASSASANVTPHHTIAQESC 825

human/1-422      --------------------------------------------------
fly/1-898    826 PSPCSSASHFGVAHSSGFSSDPISPAVSSYAHMSYNYASSANTMTPSSAS 875

human/1-422      -----------------------
fly/1-898    876 GTSAHVAPGKQQFFASCFYSPWV                            898
```

Alignment of an orthologous protein in *D.melanogaster vs H.sapiens*

Not suitable for global alignment

2 main regions of similarity

Better to use local alignment

Local alignment – Smith-Waterman algorithm

Example – align TCGA to GAC

0	-	T	C	G	A
-	0	0	0	0	0
G	0	-1	-1	1	-1
A	0	-1	-2	-1	2
C	0	-1	0	-2	0

Points for match	= +1
Points for mismatch	= -1
Points for a gap insertion	= -2

Local alignment – Smith-Waterman algorithm

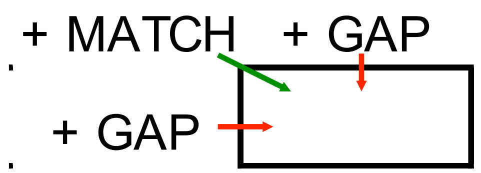
Example – align TCGA to GAC

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

Points for match	= +1
Points for mismatch	= -1
Points for a gap insertion	= -2

Local alignment – Smith-Waterman algorithm

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



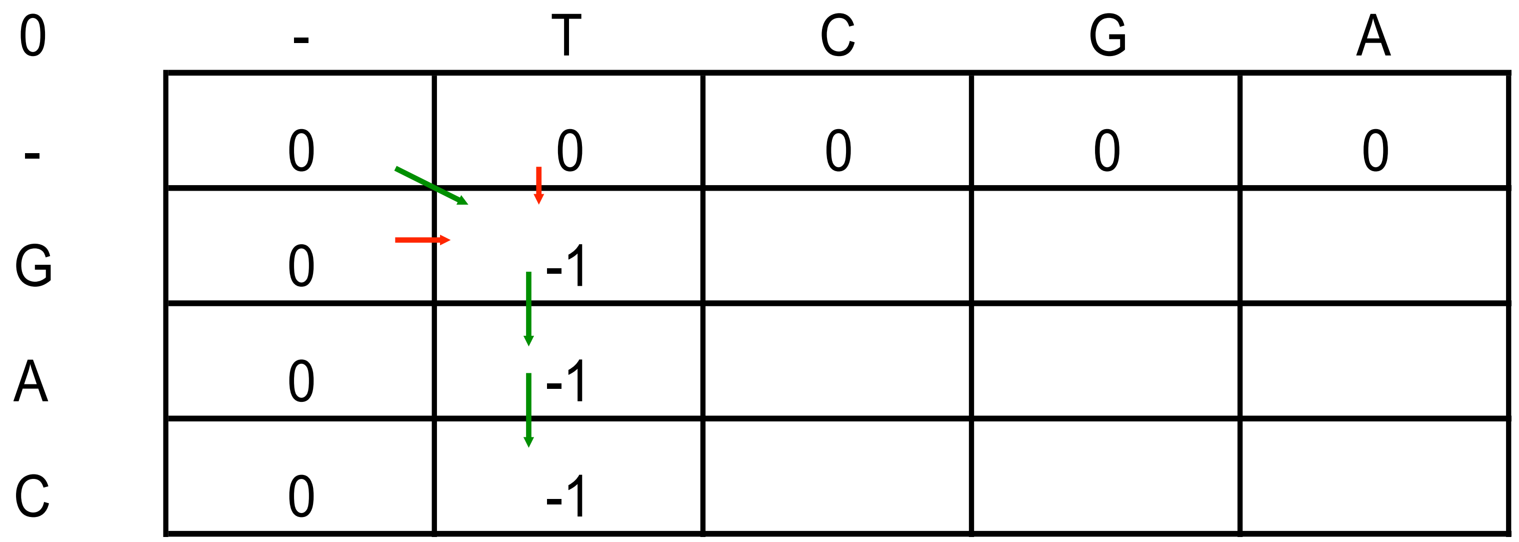
Points for match	= +1
Points for mismatch	= -1
Points for a gap insertion	= -2

Local alignment – Smith-Waterman algorithm

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

Points for match	= +1
Points for mismatch	= -1
Points for a gap insertion	= -2

Local alignment – Smith-Waterman algorithm

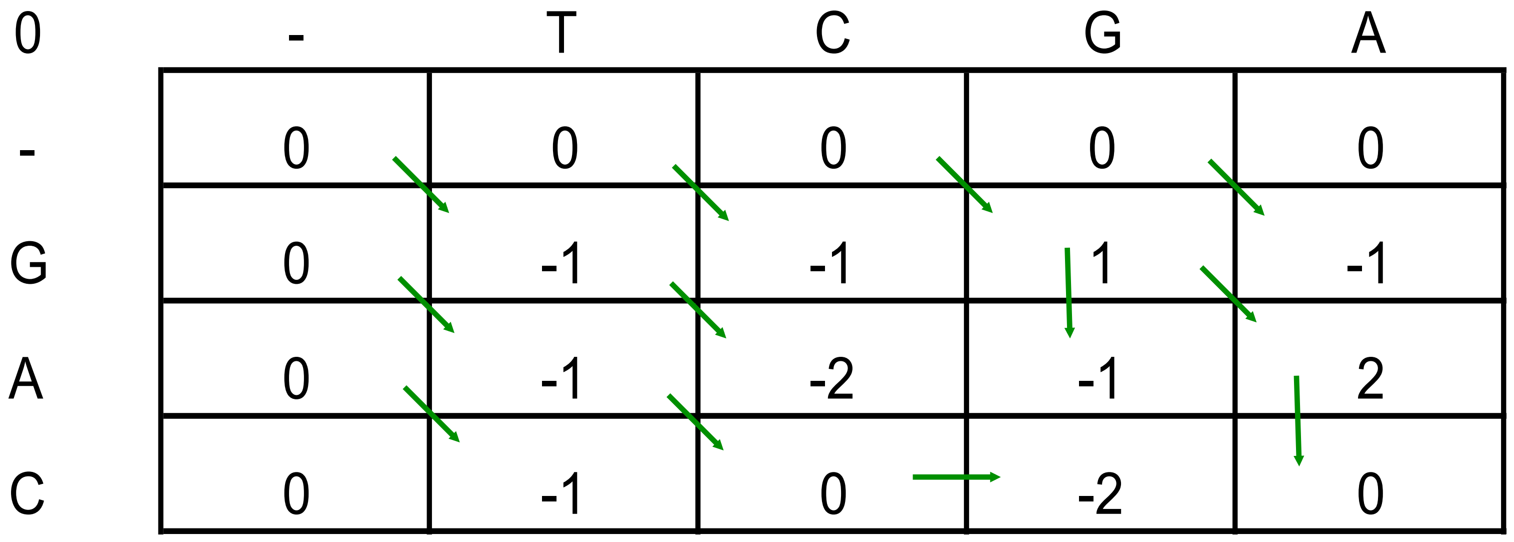


0	-	T	C	G	A
-	0	0	0	0	0
G	0	-1			
A	0	-1			
C	0	-1			

Points for match	= +1
Points for mismatch	= -1
Points for a gap insertion	= -2

Local alignment – Smith-Waterman algorithm

Example – align TCGA to GAC



0	-	T	C	G	A
-	0	0	0	0	0
G	0	-1	-1	1	-1
A	0	-1	-2	-1	2
C	0	-1	0	-2	0

Points for match	= +1
Points for mismatch	= -1
Points for a gap insertion	= -2

Backtracking and final local alignment

0	-	T	C	G	A
-	0	0	0	0	0
G	0	-1	-1	1	-1
A	0	-1	-2	-1	2
C	0	-1	0	-2	0

```
GA
| |
GA
```

Smith-Waterman – more details

http://www.youtube.com/watch?v=IVRSFaGCGeE

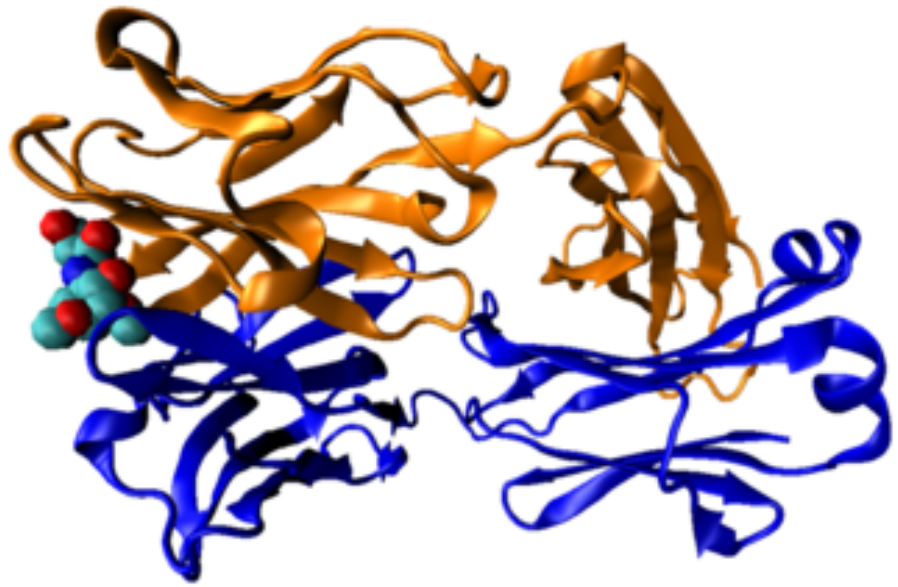
Dynamic programming

- Needleman-Wunsch and Smith-Waterman are a class of methods known as ‘Dynamic Programming’
- Guaranteed to give you the best possible alignment
- In biology, this algorithm is very inefficient because any 2 randomly selected DNA fragments in a database are are unlikely to have any similarity
- Therefore, these methods take a long time to run

BLAST –
Basic Local Alignment Search Tool

Background – BLAST

- Primarily designed to identify homologous sequences
  - Blast is a hashed seed-extend algorithm
  - Negative selection
  - Only some parts of a sequence are usually constrained



BLAST - Original version

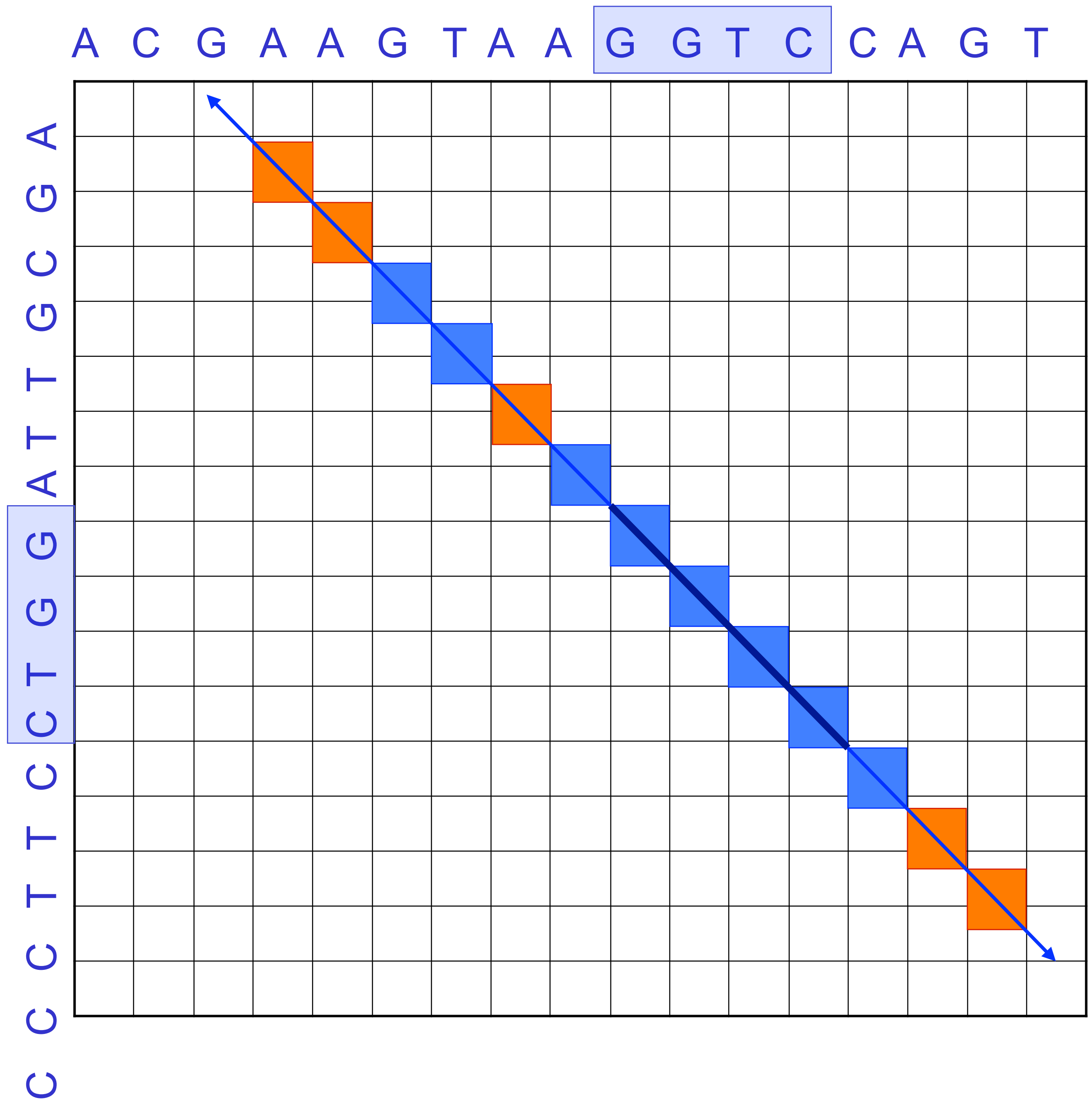
Example:

Seed size = 4,
No mismatches in seed

The matching word GGTC
initiates an alignment

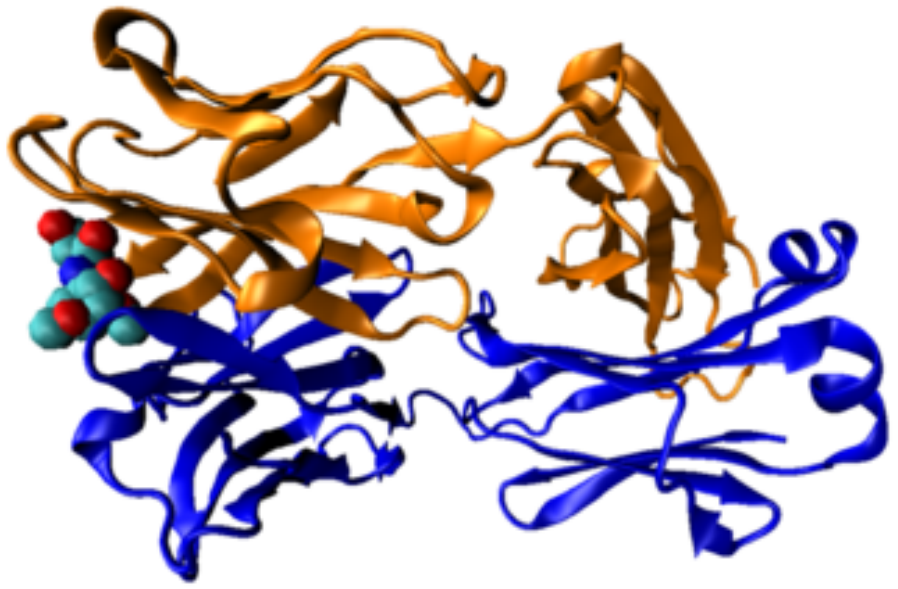
Extension to the left and right
with no gaps until alignment
score falls below 50%

Output:
GTAAGGTCC
GTTAGGTCC


A C G A A G T A A G G T C C A G T
A G C G T A G G T C C T T C C

BLAST - Original algorithm

- Finding seeds significantly increases the speed of BLAST compared to doing a full local alignment over a whole sequence
- Will not guarantee the best solution
- BLAST first finds highly conserved or identical sequences which are then extended with a local alignment.



BLAST – Speed (or lack thereof)

- Typically BLAST will take approximately 0.1 – 1 second to search 1 sequence against a database
- Depends on size of database, e-value cutoff and number of hits to report selected
- 60 million reads equates to 70 CPU days!
- Even on multi-core systems this is too long!
- Especially if you have multiple samples!
- This is still true of FPGA and SIMD (vectorised) implementations of BLAST

When NOT to use *BLAST*

- A typical situation: you have lots DNA sequences and want to extend it or find where on a genome it maps.
- In other words, you want an **exact** or **near-exact** match to a sequence that is part of an **assembled genome**.
- Short reads require very fast algorithms for finding near-exact matches in genomic sequences:
  - BLAT
    - Highly recommended: the BLAT paper (Kent WJ (2003) *Genome Res* 12:656-64) – very well written
  - SOAP
  - Bowtie/Bowtie 2
  - MAQ
  - BWA
  - Shrimp2

Contents

- **Alignment algorithms for short-reads**
  - Background – Blast (why can’t we use it?)
  - **Adapting hashed seed-extend algorithms to work with shorter reads**
  - Indel detection
  - Suffix/Prefix Tries
  - Other alignment considerations
  - Typical alignment pipeline
  - New methods of variant calling

Adapting hashed seed-extend algorithms to work with shorter reads

- Improve seed matching sensitivity
  - Allow mismatches within seed
    - BLAST
  - Allow mismatches + Adopt spaced-seed approach
    - ELAND, SOAP, MAQ, RMAP, ZOOM
  - Allow mismatches + Spaced-seeds + Multi-seeds
    - SSAHA2, BLAT, ELAND2
- Above and/or Improve speed of local alignment for seed extension
  - Single Instruction Multiple Data
    - Shrimp2, CLCBio
  - Reduce search space to region around seed

Hashed seed-extend algorithms

- These are most similar to BLAST
- Are not designed to work with large databases

- **2 step process**
  - Identify a match to the seed sequence in the reference
  - Extend match using sensitive (but slow) Smith-Waterman algorithm (dynamic programming)

Seed-extend algorithm

Reference sequence:

...ACTGGGTCATCGTACGATCGATCGATCGATCGATCGGCTAGCTAGCTA...

Short read:

GTCATCGTACGATCGATAGATCGATCGATCGGCTA

Note that the short read has 1 difference wrt to reference

Seed-extend algorithm

Reference sequence:

...ACTGGGTCATCGTACGATCGATCGATCGATCGATCGGCTAGCTAGCTA...

Short read:

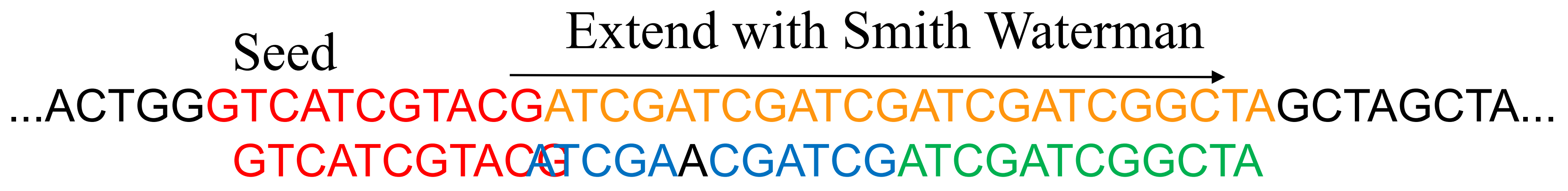
GTCATCGTACG ATCGATAGATCG ATCGATCGGCTA

11bp word 11bp word 11bp word

The algorithm will try to match each word to the reference. If there is a match at with any single word it will perform a local alignment to extend the match

Seed-extend algorithm

Reference sequence:



Seed
Extend with Smith Waterman
...ACTGGGTCATCGTACGATCGATCGATCGATCGATCGGCTAGCTAGCTA...
GTCATCGTACGATCGAACGATCGATCGATCGGCTA

Short read:

GTCATCGTACG ATCGATAGATCG ATCGATCGGCTA

Here the algorithm is able to match the short read with a word length of 11bp

Seed-extend algorithm

Reference sequence:

...ACTGGGTCATCGTACGATCGATCGATCGATCGATCGGCTAGCTAGCTA...

Short read:

GTCATCGTACGATCGATCGATCGATCGATCGGCAA

Note that the short read has 3 differences
Possibly sequencing errors, possibly SNPs

Seed-extend algorithm

Reference sequence:

...ACTGGGTCATCGTACGATCGATCGATCGATCGATCGGCTAGCTAGCTA...

Short read:

GTCATCGTACG ATCGATCGATCG ATCGATCGGCAA

11bp word 11bp word 11bp word

Note that the short read has 3 differences

Seed-extend algorithm

Reference sequence:

...ACTGGGTCATCGTACGATCGATCGATCGATCGATCGGCTAGCTAGCTA...

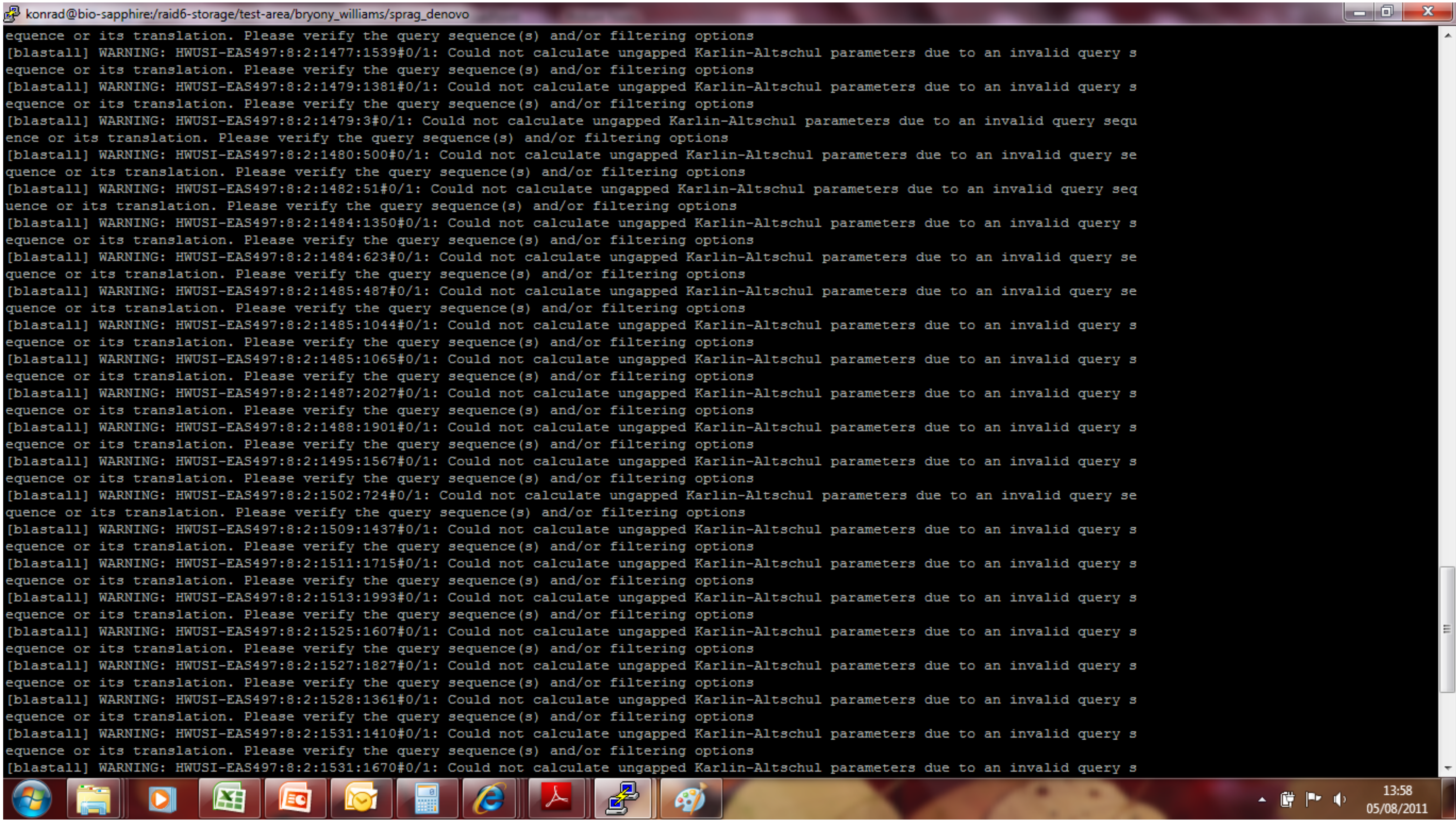
Short read:

GTCATCGTACG ATCGATCGATCG

ATCGATCGGCAA

No seeds match

Therefore the algorithm would find no hits at all!



konrad@bio-sapphire:/raid6-storage/test-area/bryony_williams/sprag_denovo

```
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1477:1539#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1479:1381#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1479:3#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query sequ
ence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1480:500#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query se
quence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1482:51#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query seq
uence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1484:1350#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1484:623#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query se
quence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1485:487#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query se
quence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1485:1044#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1485:1065#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1487:2027#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1488:1901#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1495:1567#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1502:724#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query se
quence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1509:1437#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1511:1715#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1513:1993#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1525:1607#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1527:1827#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1528:1361#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1531:1410#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
equence or its translation. Please verify the query sequence(s) and/or filtering options
[blastall] WARNING: HWUSI-EAS497:8:2:1531:1670#0/1: Could not calculate ungapped Karlin-Altschul parameters due to an invalid query s
```

Adapting hashed seed-extend algorithms to work with shorter reads

- Improve seed matching sensitivity
  - **Allow mismatches within seed**
    - **BLAST**
  - Allow mismatches + Adopt spaced-seed approach
    - ELAND, SOAP, MAQ, RMAP, ZOOM
  - Allow mismatches + Spaced-seeds + Multi-seeds
    - SSAHA2, BLAT, ELAND2
- Above and/or Improve speed of local alignment for seed extension
  - Single Instruction Multiple Data
    - Shrimp2, CLCBio
  - Reduce search space to region around seed

Adapting hashed seed-extend algorithms to work with shorter reads

- Improve seed matching sensitivity
  - **Allow mismatches within seed**
    - **BLAST**
  - **Allow mismatches + Adopt spaced-seed approach**
    - **ELAND, MAQ, RMAP, ZOOM**
  - Allow mismatches + Spaced-seeds + Multi-seeds
    - SSAHA2, BLAT, ELAND2
- Above and/or Improve speed of local alignment for seed extension
  - Single Instruction Multiple Data
    - Shrimp2, CLCBio
  - Reduce search space to region around seed

Consecutive seed

Consecutive seed 9bp allowing no mismatches:

ACTCCCATCGTCATCGTACTAGGGATCGTAACA Reference sequence

CCACTGTCCTCCTACATAGGAACGA SNP ‘heavy’ read

TC**A**TC**G**TAC

TC**C**TC**C**TAC Cannot find seed match due to A->C SNP and G->C SNP

Even allowing for 2 mismatches in the seed - no seeds match.
No hits!

Spaced seeds

To increase sensitivity we can used spaced-seeds:

```
    111111111         Consecutive seed template with length 9bp
ACTATCATCGTACACAT     Reference
    TCATCGTAC         Query
```

```
11001100110011001     Spaced-seed template with weight 9bp
ACTATCATCGTACACAT     Reference
ACTCTCACCGTACACAT     Query
```

Spaced seeds

Spaced seed with weight 9bp and no mismatches:

ACTCCCATTGTCATCGTACTTGGGATCGTAACA Reference sequence

CCACTGTAATCGTACATGGGAACGA SNP ‘heavy’ read

CCATTGTCATCGTACAT

CCXXTGXXATXXTAXXT

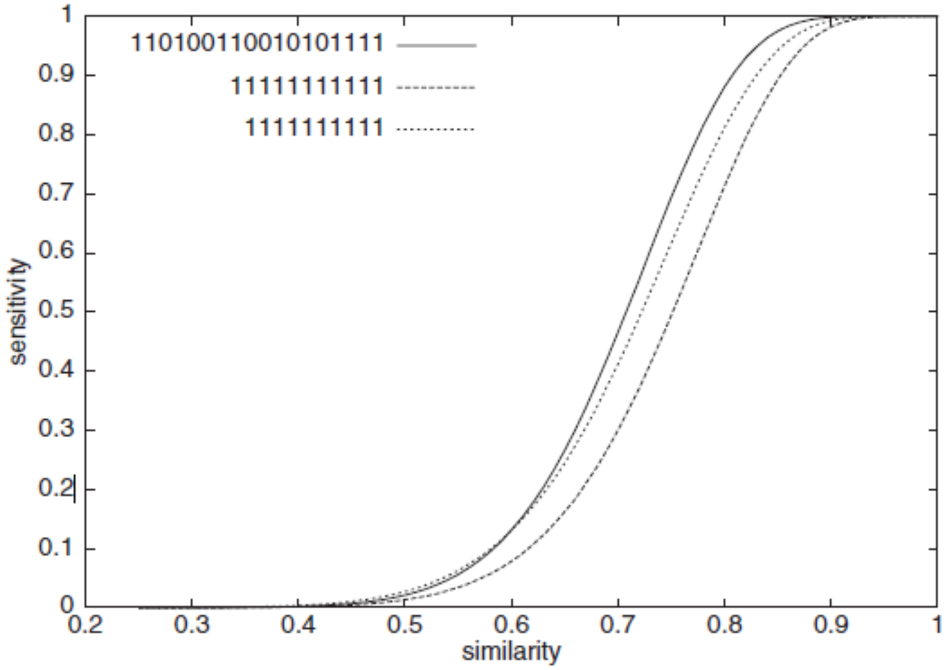
Despite SNPs – seed matched with 0 mismatches

Can now extend with Smith-Waterman or other local alignment

Spaced seeds

Spaced seeds:

• A seed template ‘111010010100110111’ is 55% more sensitive than BLAST’s default template ‘11111111111’ for two sequences of 70% similarity
• Typically seeds of length ~30bp and allow up to 2 mismatches in short read datasets



Ma, B. et al. PatternHunter. Bioinformatics Vol 18, No 3, 2002

Contents

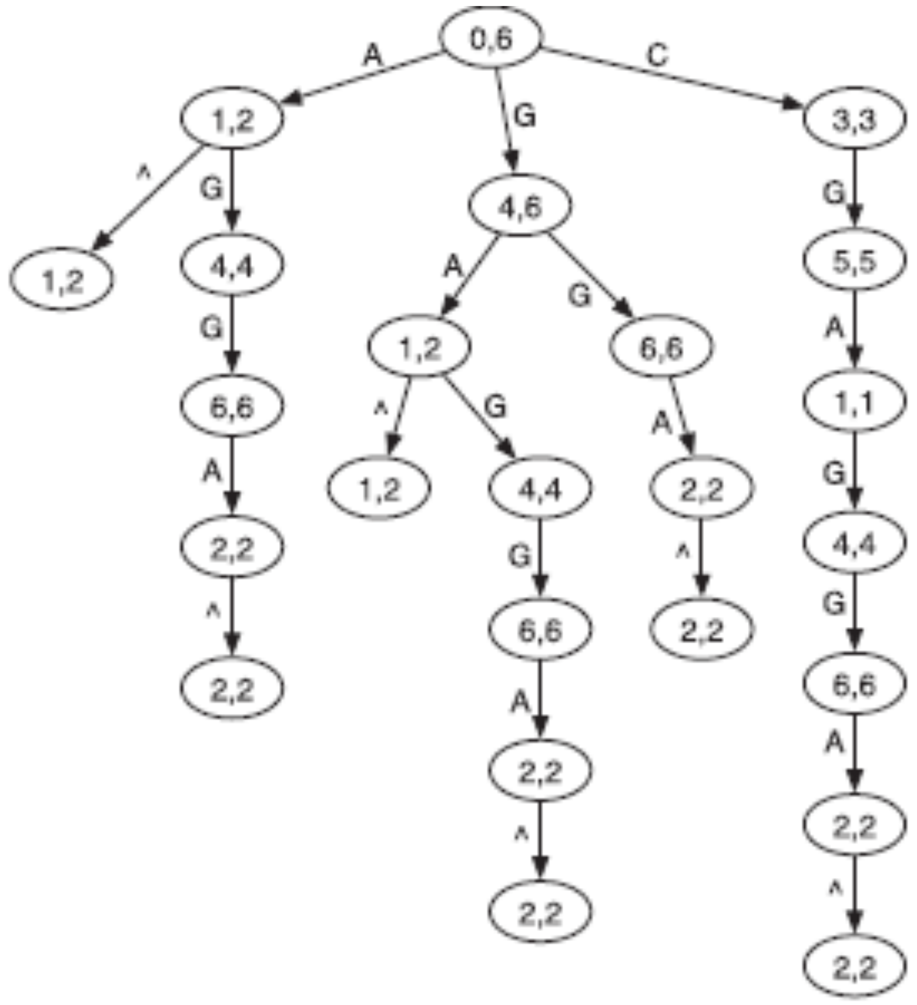
- **Alignment algorithms for short-reads**
  - Background – Blast (why can’t we use it?)
  - Adapting hashed seed-extend algorithms to work with shorter reads
  - **Suffix/Prefix Tries**
  - Indel detection
  - Other alignment considerations
  - Typical alignment pipeline
  - New methods of SNP calling

Suffix-Prefix Trie

- Trie – data structure which stores the suffixes (i.e. ends of a sequence)
- A family of methods which uses a Trie structure to search a reference sequence
  - Bowtie
  - BWA aln (<70bp reads) and MEM algorithm (>70bp reads)
  - SOAP version 2
- Key advantages:
  - 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 memory requirements (e.g. Human genome can be stored in 2Gb of RAM)
  - Burrows Wheeler Transform to perform fast lookups

Suffix Trie

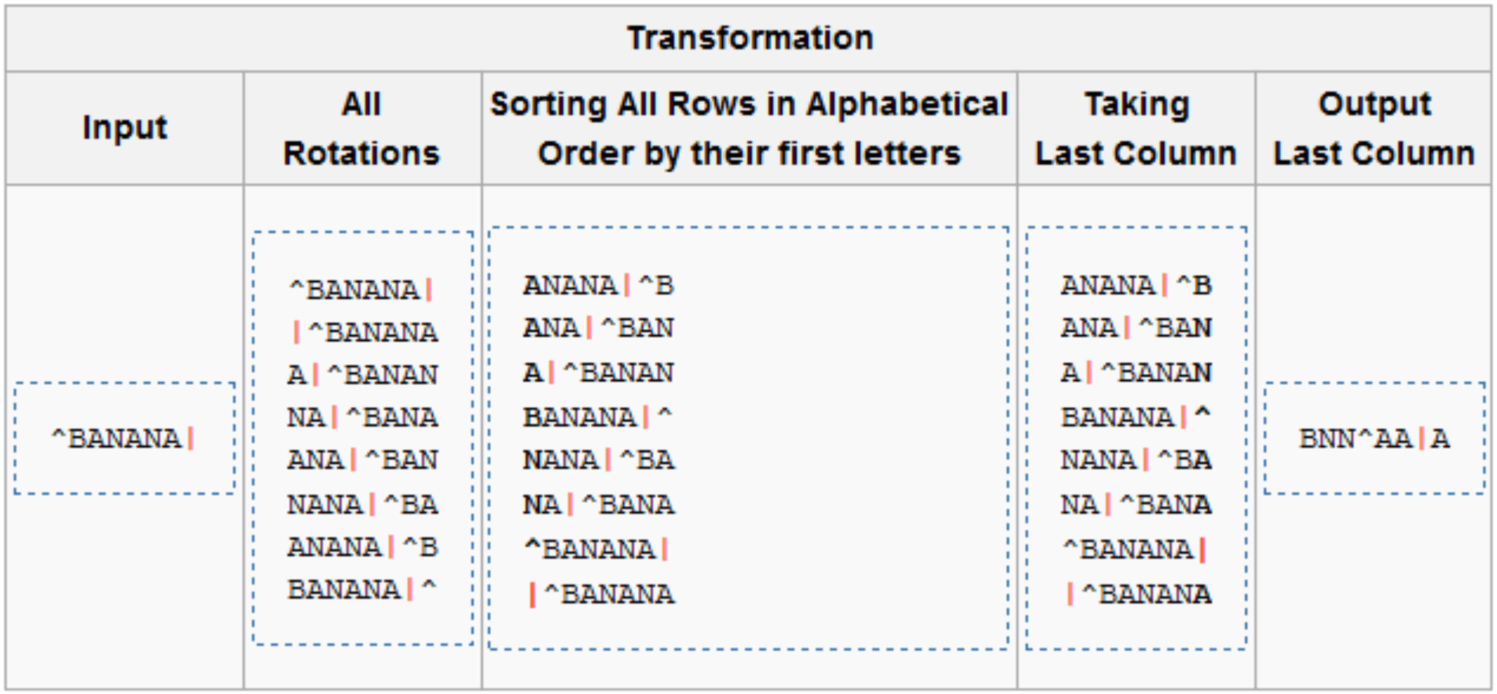
Read
AGGAGC



Heng Li & Nils Homer. Sequence alignment algorithms for next-generation sequencing. Briefings in Bioinformatics. Vol 11. No 5. 473 483, 2010

Burrows-Wheeler Algorithm

- Encodes data so that it is easier to compress
- Burrows-Wheeler transform of the word BANANA
- Can later be reversed to recover the original word



Transformation				
Input	All Rotations	Sorting All Rows in Alphabetical Order by their first letters	Taking Last Column	Output Last Column
^BANANA|	^BANANA| |^BANANA A|^BANAN NA|^BANA ANA|^BAN NANA|^BA ANANA|^B BANANA|^	ANANA|^B ANA|^BAN A|^BANAN BANANA|^ NANA|^BA NA|^BANA ^BANANA| |^BANANA	ANANA|^B ANA|^BAN A|^BANAN BANANA|^ NANA|^BA NA|^BANA ^BANANA| |^BANANA	BNN^AA|A

More Burrows-Wheeler

Input	SIX.MIXED.PIXIES.SIFT.SIXTY.PIXIE.DUST.BOXES
Burrows-Wheeler Output	TEXYDST.E.IXIXIXXSSMPPS.B..E.S.EUSFXDIIOIIIT

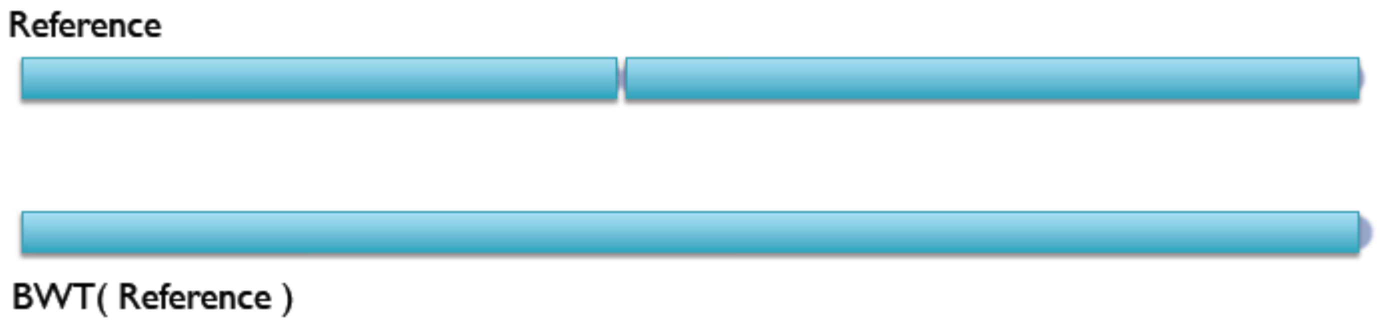
Repeated characters mean that it is easier to compress

Suffix Trie for a bacterial genome would be > 1Tb

We have to compress it

Use FM-Index/BW transform to do this compression

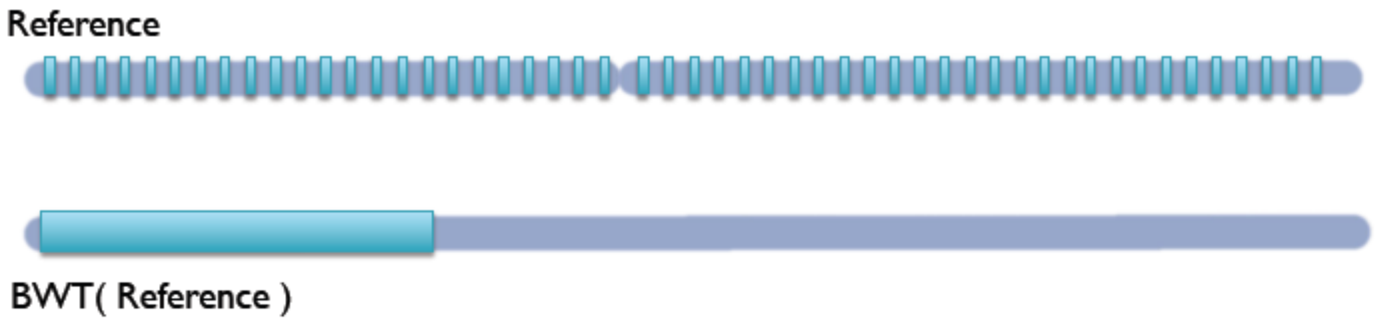
Bowtie/BWA example



Query:
AATGATACGGCGACCACCGAGATCTA

Courtesy Mike Schatz

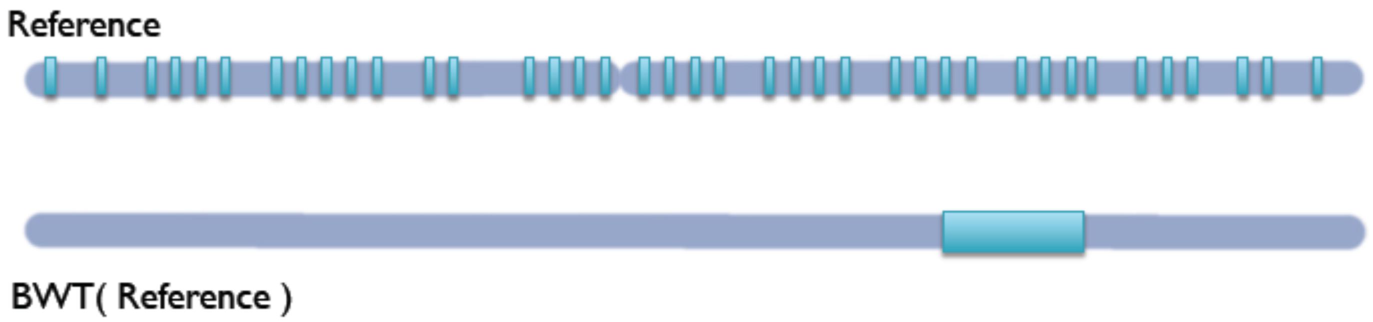
Bowtie/BWA example



Query:
AATGATACGGCGACCACCGAGATCT$\boxed{\text{A}}$

Courtesy Mike Schatz

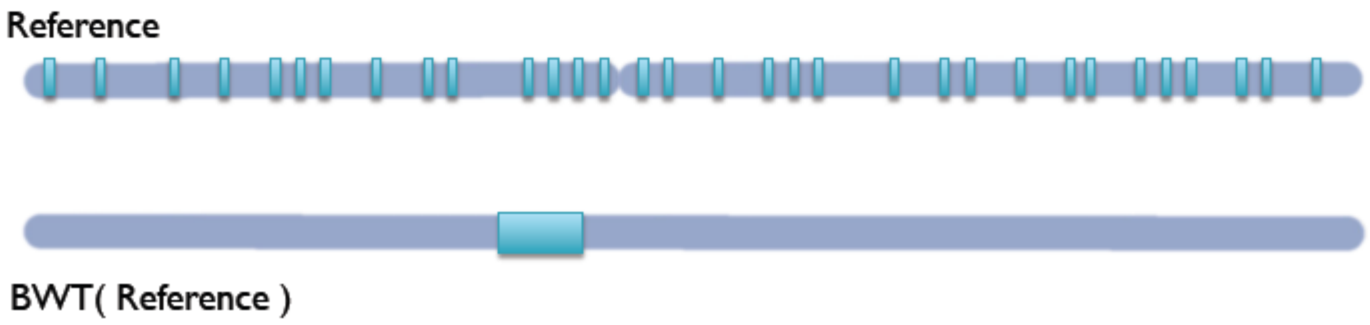
Bowtie/BWA example



Query:
AATGATACGGCGACCACCGAGATC$\boxed{\text{TA}}$

Courtesy Mike Schatz

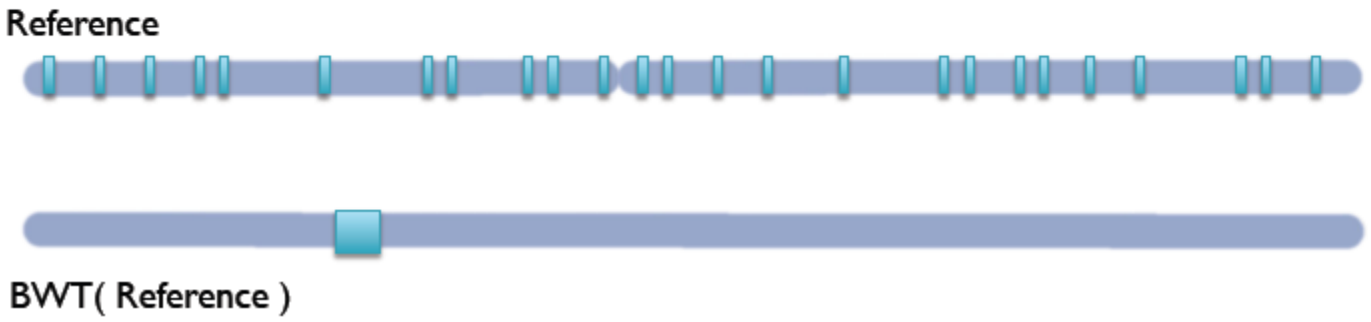
Bowtie/BWA example



Query:
AATGATACGGCGACCACCGAGAT$\boxed{\text{CTA}}$

Courtesy Mike Schatz

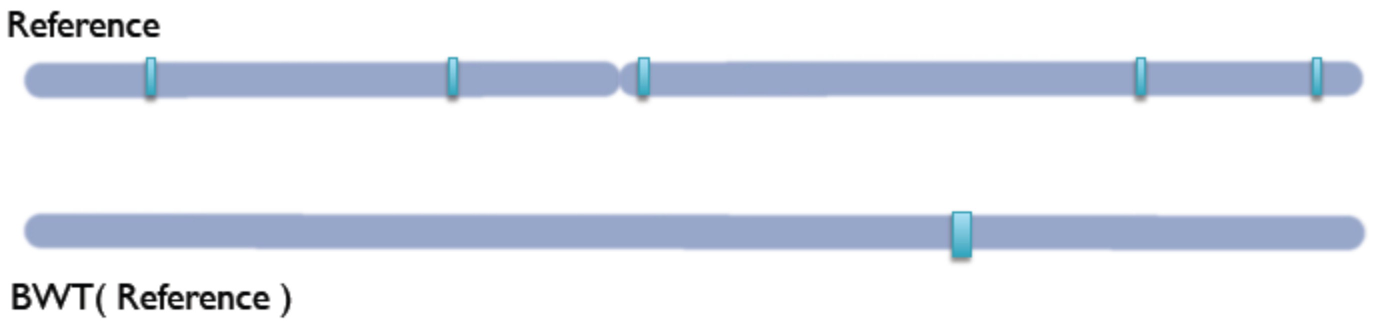
Bowtie/BWA example



Query:
AATGATACGGCGACCACCGAGATCTA

Courtesy Mike Schatz

Bowtie/BWA example

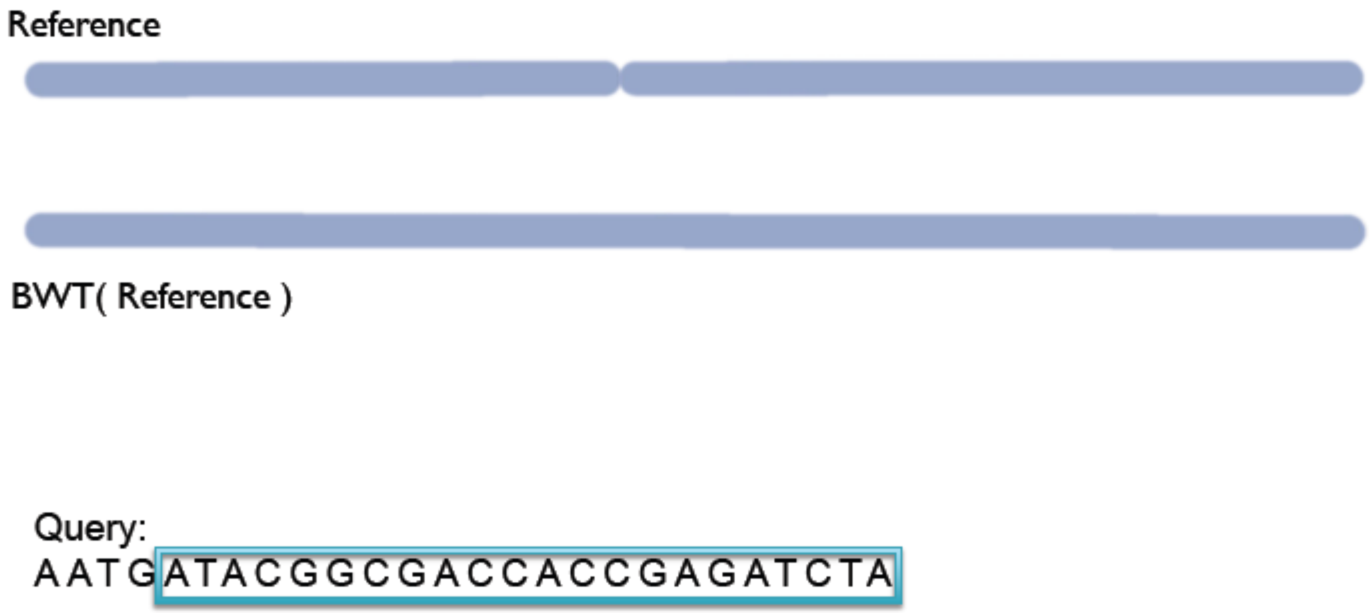


Query:

AATGATACGGCGACCACCGAGATCTA

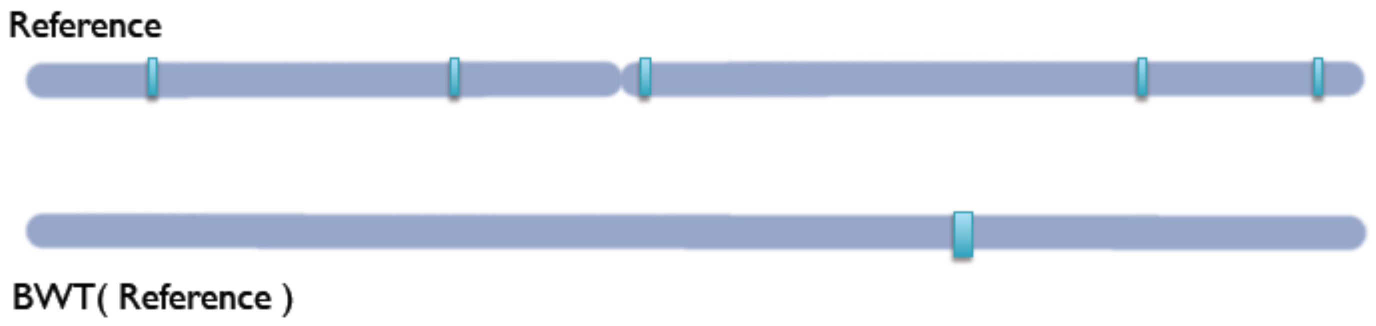
Courtesy Mike Schatz

Bowtie/BWA example



Courtesy Mike Schatz

Bowtie/BWA example



Query:
AATGTTACGGCGACCACCGAGATCTA

Courtesy Mike Schatz

Bowtie/BWA example

Reference
BWT(Reference)

Query:

AATGTTACGGCGACCACCGAGATCTA

Courtesy Mike Schatz

Bowtie/Soap2 vs. BWA

- Bowtie 1 and Soap2 cannot handle gapped alignments
  - No indel detection => Many false SNP calls

Bowtie/Soap2:

```
ACTCCCATTGTCATCGTACTTGGGATCGTAACA Reference
    CCATTGTCATCGTACTTGGGATCTA
          TCATCGTACTTGGGATCTA
                   TTGGGATCTA
```

False SNPs

N.B. Bowtie2 can handle gapped alignments

Bowtie/Soap2 vs. BWA

- Bowtie 1 and Soap2 cannot handle gapped alignments
  - No indel detection => Many false SNP calls

BWA:

```
ACTCCCATTGTCATCGTACTTGGGATCGTAACA Reference
    CCATTGTCATCGTACTTGGGATC-TA
          TCATCGTACTTGGGATC-TA
                   TTGGGATC-TA
```

N.B. Bowtie2 can handle gapped alignments

Comparison

Hash referenced spaced seeds

- Requires ~50Gb of memory
- Runs 30-fold slower
- Is much simpler to program
- Most sensitive

Indexed Suffix/Prefix Trie

- Requires <2Gb of memory
- Runs 30-fold faster
- Is much more complicated to program
- Least sensitive

There are limits however

With longer 100-300 bp reads, multiple indels or variable regions longer than a few bp are likely to be missed

```
ACTCCCATTGTCATCGTACTTGGGATCGTAACA Reference
    CCATTGTCAACCATCTAGTAGCT-TA
          TCAACCATCTAGTAGCT-TA
                    ACCATCTA-TA
```

You only find what you are looking for

- What happens if there are SNPs and Indels in the same region?

Let’s assume that the SNP caller made this call of a single SNP:

```
ATGTATGTA
ATGTGTGTA
```

and the indel caller produced this call of a 3 base deletion:

```
ATGTATGTA
ATGT---TA
```

Should we assume this is a heterozygous SNP opposite a heterozygous Indel or a more complex locus?

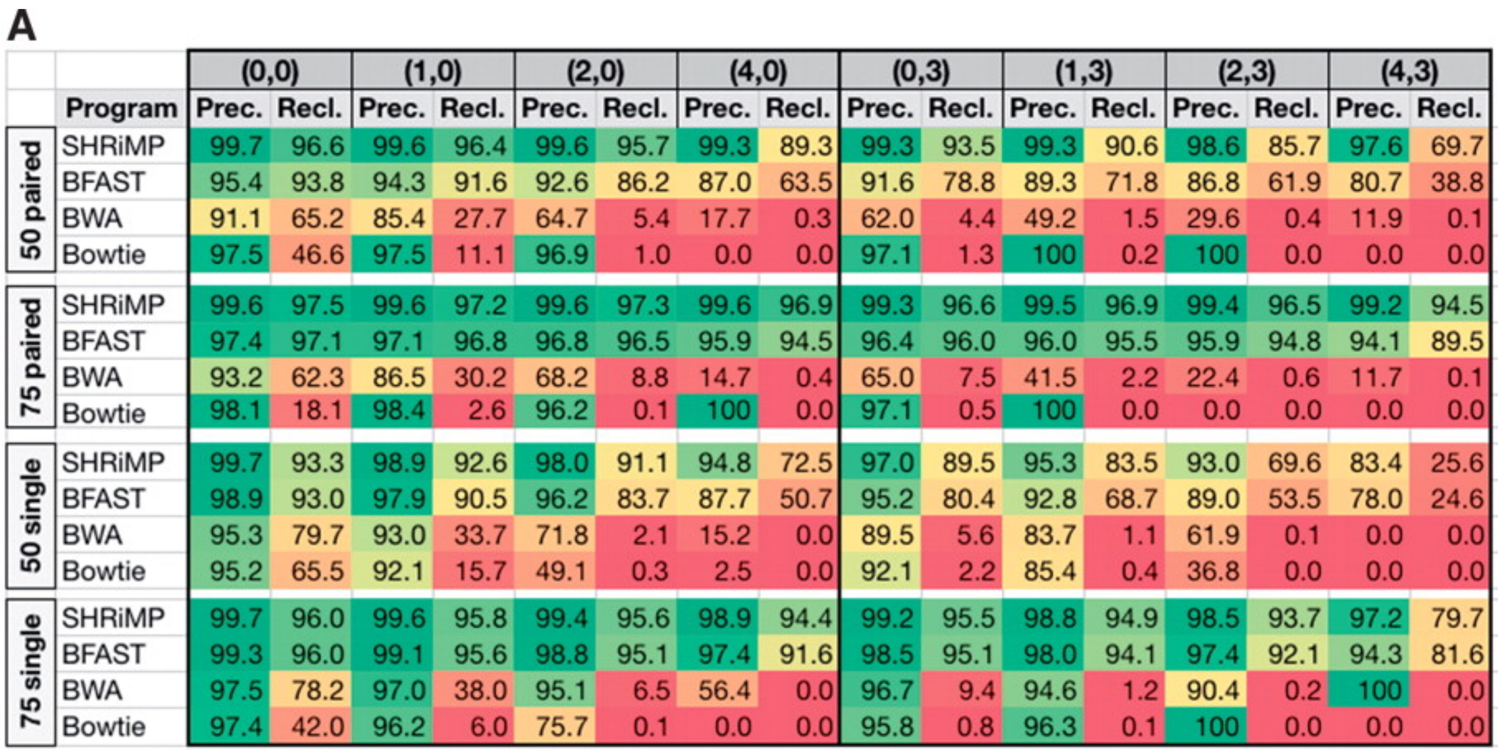
Comparison

- Bowtie's reported 30-fold speed increase over hash-based methods with small loss in sensitivity
- Limitations to Trie-based approaches:
  - Only able to find alignments within a certain 'edit distance’
  - Important to quality clip reads (-q in BWA)
  - Non-A/C/G/T bases on reads are often treated as mismatches
  - Make sure Ns are removed!

Hash based approaches are more suitable for divergent alignments

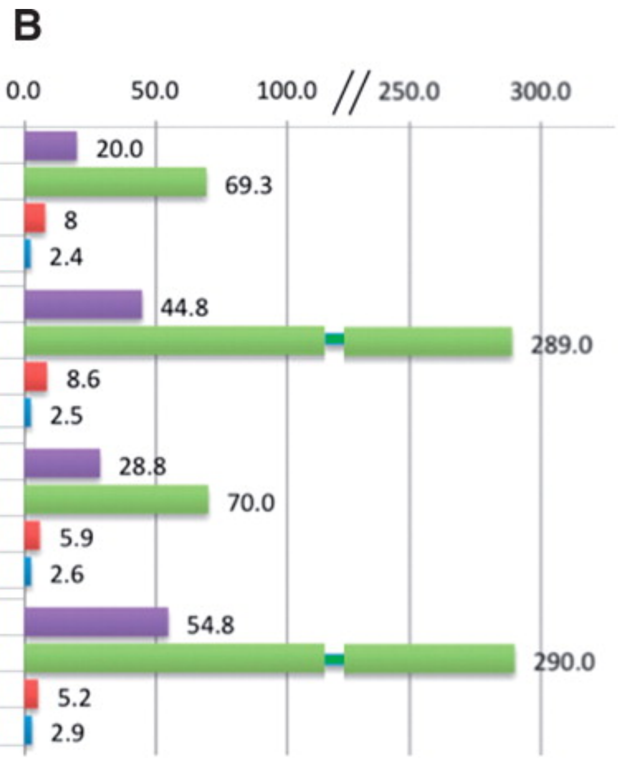
- Rule of thumb:
  - <2% divergence -> Trie-based
    - E.g. human alignments
  - >2% divergence -> Seed-extend based approach
    - E.g. wild mouse strain alignments

Precision and recall by amount of variation for 4 datasets, by polymorphism: (number of SNPs, Indel size)



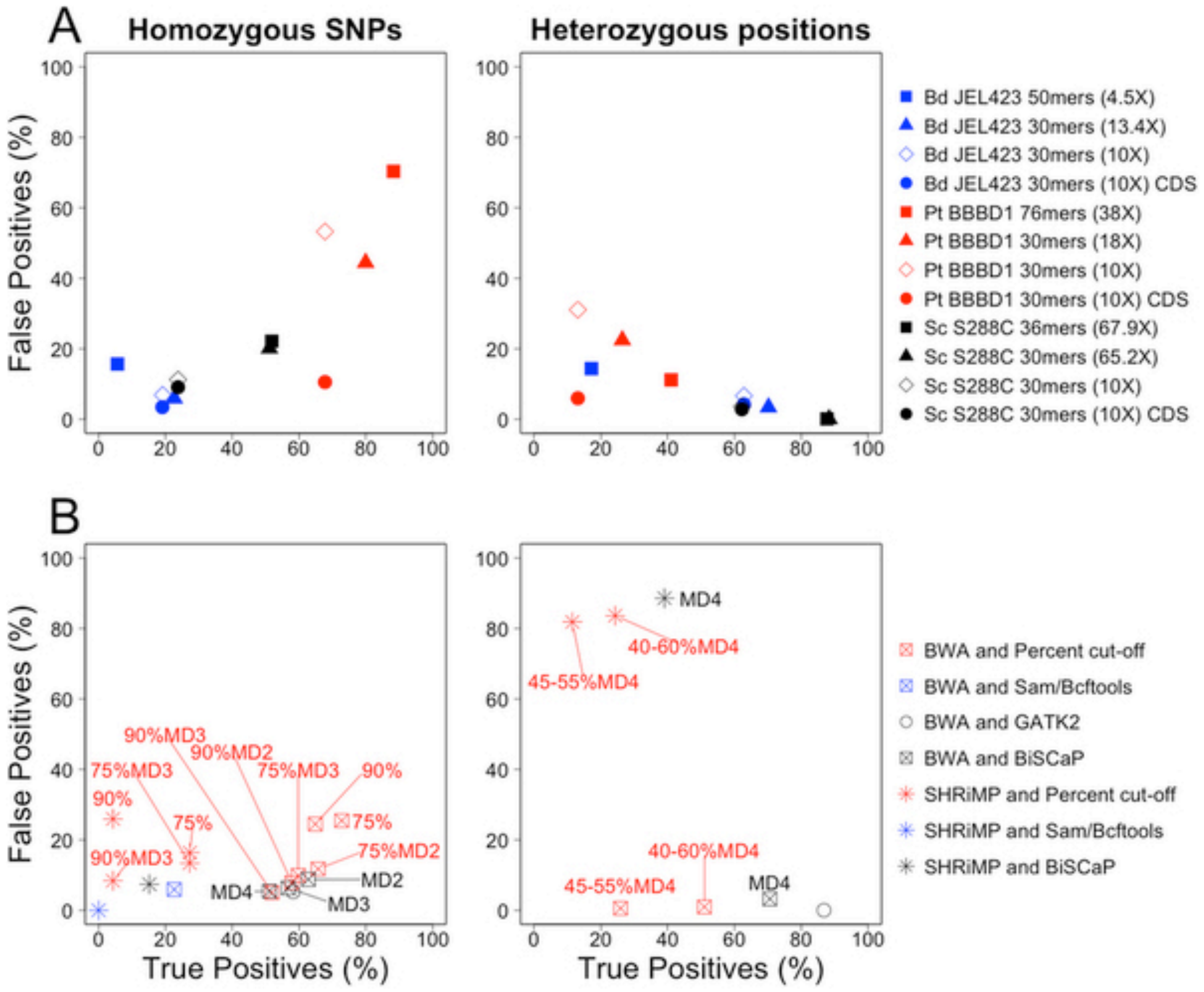
A

		(0,0)		(1,0)		(2,0)		(4,0)		(0,3)		(1,3)		(2,3)		(4,3)	
	Program	Prec.	Recl.	Prec.	Recl.	Prec.	Recl.	Prec.	Recl.	Prec.	Recl.	Prec.	Recl.	Prec.	Recl.	Prec.	Recl.
50 paired	SHRiMP	99.7	96.6	99.6	96.4	99.6	95.7	99.3	89.3	99.3	93.5	99.3	90.6	98.6	85.7	97.6	69.7
	BFAST	95.4	93.8	94.3	91.6	92.6	86.2	87.0	63.5	91.6	78.8	89.3	71.8	86.8	61.9	80.7	38.8
	BWA	91.1	65.2	85.4	27.7	64.7	5.4	17.7	0.3	62.0	4.4	49.2	1.5	29.6	0.4	11.9	0.1
	Bowtie	97.5	46.6	97.5	11.1	96.9	1.0	0.0	0.0	97.1	1.3	100	0.2	100	0.0	0.0	0.0
75 paired	SHRiMP	99.6	97.5	99.6	97.2	99.6	97.3	99.6	96.9	99.3	96.6	99.5	96.9	99.4	96.5	99.2	94.5
	BFAST	97.4	97.1	97.1	96.8	96.8	96.5	95.9	94.5	96.4	96.0	96.0	95.5	95.9	94.8	94.1	89.5
	BWA	93.2	62.3	86.5	30.2	68.2	8.8	14.7	0.4	65.0	7.5	41.5	2.2	22.4	0.6	11.7	0.1
	Bowtie	98.1	18.1	98.4	2.6	96.2	0.1	100	0.0	97.1	0.5	100	0.0	0.0	0.0	0.0	0.0
50 single	SHRiMP	99.7	93.3	98.9	92.6	98.0	91.1	94.8	72.5	97.0	89.5	95.3	83.5	93.0	69.6	83.4	25.6
	BFAST	98.9	93.0	97.9	90.5	96.2	83.7	87.7	50.7	95.2	80.4	92.8	68.7	89.0	53.5	78.0	24.6
	BWA	95.3	79.7	93.0	33.7	71.8	2.1	15.2	0.0	89.5	5.6	83.7	1.1	61.9	0.1	0.0	0.0
	Bowtie	95.2	65.5	92.1	15.7	49.1	0.3	2.5	0.0	92.1	2.2	85.4	0.4	36.8	0.0	0.0	0.0
75 single	SHRiMP	99.7	96.0	99.6	95.8	99.4	95.6	98.9	94.4	99.2	95.5	98.8	94.9	98.5	93.7	97.2	79.7
	BFAST	99.3	96.0	99.1	95.6	98.8	95.1	97.4	91.6	98.5	95.1	98.0	94.1	97.4	92.1	94.3	81.6
	BWA	97.5	78.2	97.0	38.0	95.1	6.5	56.4	0.0	96.7	9.4	94.6	1.2	90.4	0.2	100	0.0
	Bowtie	97.4	42.0	96.2	6.0	75.7	0.1	0.0	0.0	95.8	0.8	96.3	0.1	100	0.0	0.0	0.0



David M et al. Bioinformatics 2011;27:1011-1012

False discovery rates for variants were ascertained using cFDR for three fungal NGS datasets



http://www.nature.com/srep/2013/130321/srep01512/full/srep01512.html

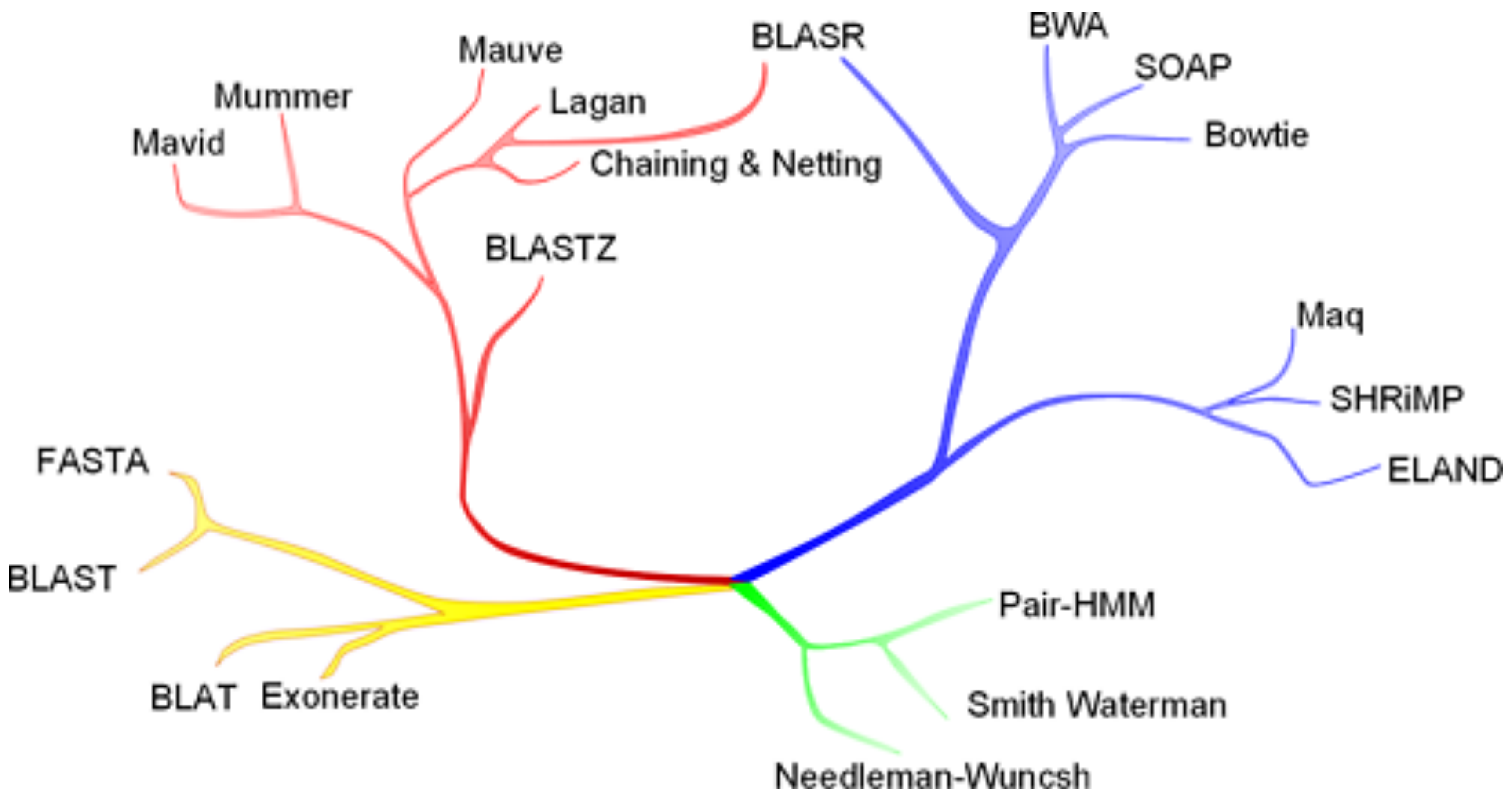
Summary of open-source short read alignment programs

Program	Algorithm	SoLID	Long reads	Gapped alignment	Paired-end	Quality scores used?
Bfast	Hashing ref	Yes	No	Yes	Yes	No
Bowtie2*	FM-Index	Yes	Yes	Yes	Yes	Yes
Blat	Hashing ref	No	Yes	Yes	No	No
BWA	FM-Index	Yes	Yes	Yes	Yes	No
MAQ	Hashing reads	Yes	No	Yes	Yes	Yes
Mosaik	Hashing ref	Yes	Yes	Yes	Yes	No
Novoalign	Hashing ref	No	No	Yes	Yes	Yes
Shrimp2	Hashing ref	Yes	Yes	Yes	Yes	Yes
SOAP2	FM-Index	No	No	No	Yes	Yes
SSAHA2	Hashing ref.	No	No	No	Yes	Yes

Heng Li & Nils Homer. Sequence alignment algorithms for next-generation sequencing. Briefings in Bioinformatics. Vol 11. No 5. 473 483, 2010

* Bowtie1 does not support gapped alignments

Aligner phylogeny



Whole genome
Pairwise heuristic
Large data set aligners
Sensitive global aligners

Sequence read aligners I use

- Genomic alignments BWA-Mem
  - Scales well with read lengths and will tolerate more errors as read lengths increase
- Bowtie2/Tophat for RNA-seq alignment
  - Splice aware and fits into a nice eco-system of tools to perform abundance expression (Cufflinks) and visualisation (cummeRbund)
- BLASR
  - Designed for PacBio reads

Alignment format for short reads – Sequence AlignMent (SAM format)

- Plain text format – human readable (sort-of)
- Eleven mandatory fields and a variable amount of optional fields.
- The optional fields are a key-value pair of TAG:TYPE:VALUE. These store extra information
- Can be converted to Binary AlignMent format (BAM) to save space and speed up look-up operations using SAMTools

Alignment format for short reads – Sequence AlignMent (SAM format)

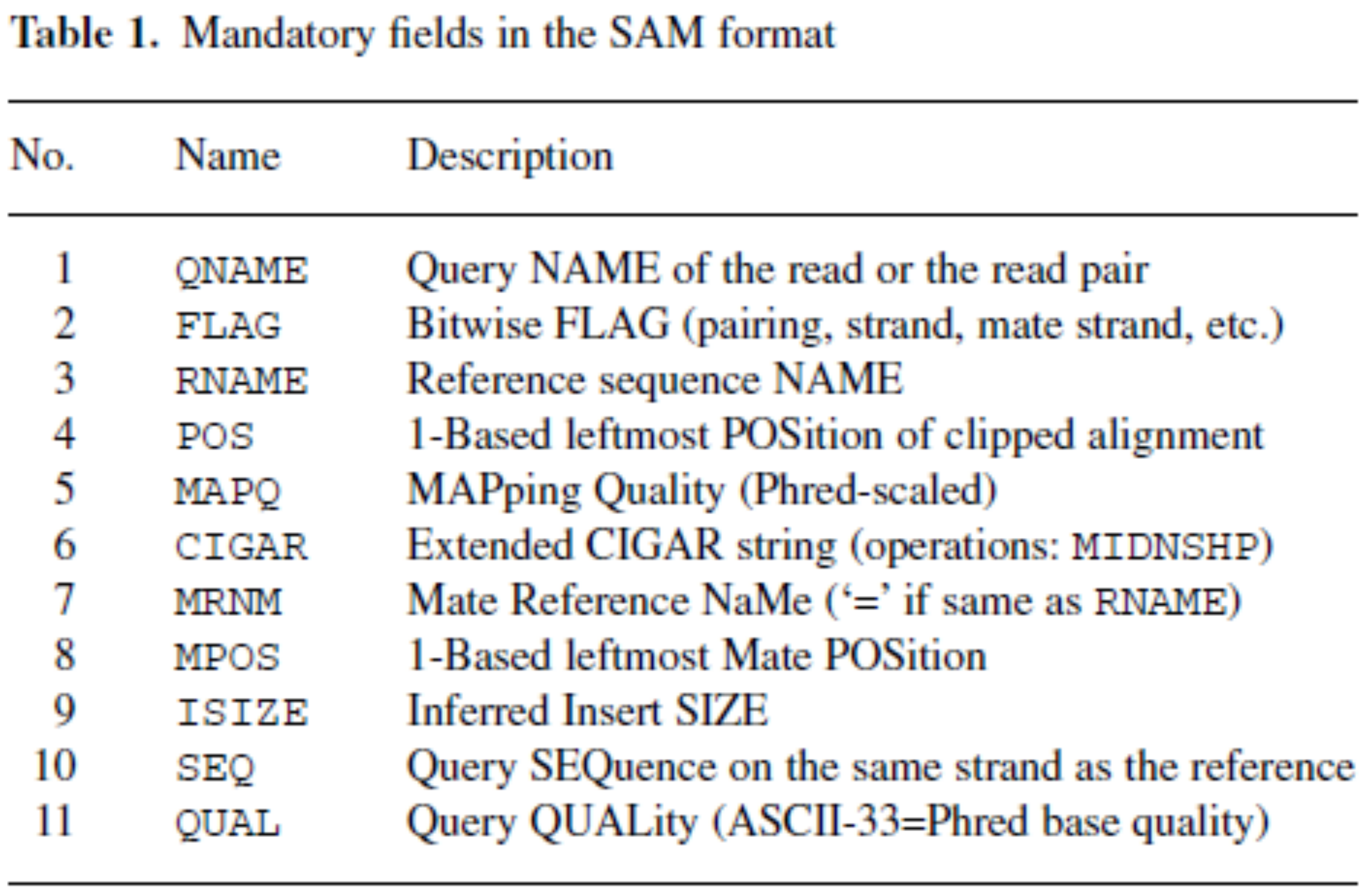
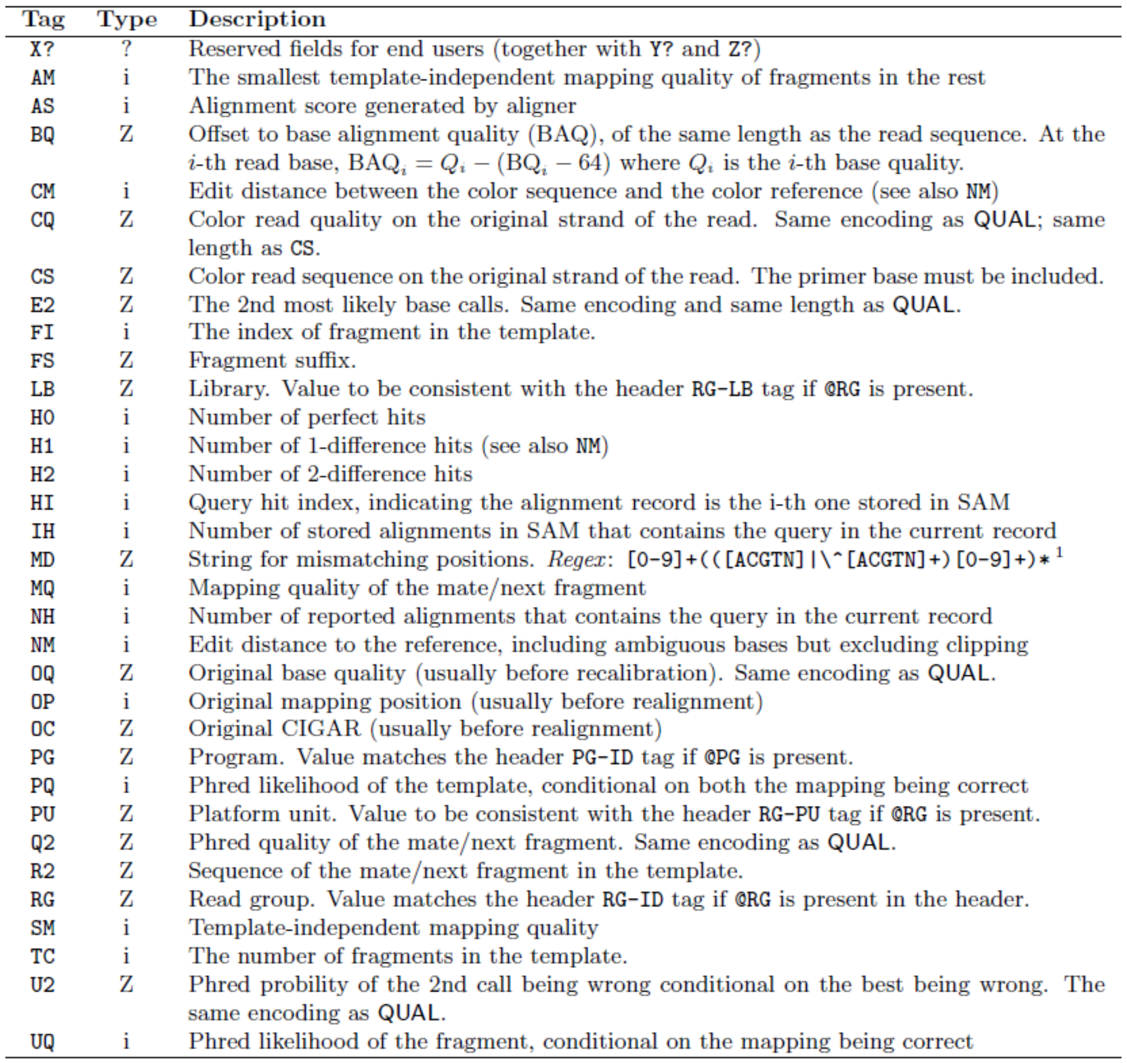


Table 1. Mandatory fields in the SAM format

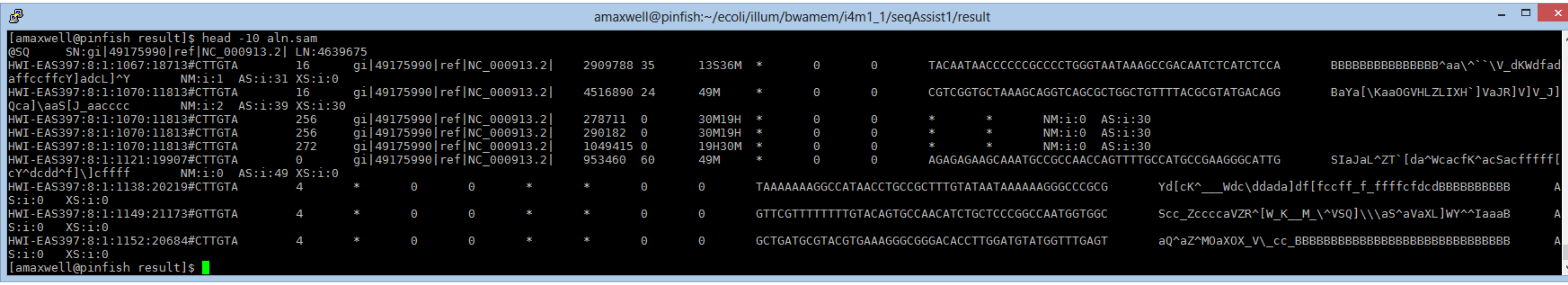
No.	Name	Description
1	`QNAME`	Query NAME of the read or the read pair
2	`FLAG`	Bitwise FLAG (pairing, strand, mate strand, etc.)
3	`RNAME`	Reference sequence NAME
4	`POS`	1-Based leftmost POSition of clipped alignment
5	`MAPQ`	MAPping Quality (Phred-scaled)
6	`CIGAR`	Extended CIGAR string (operations: `MIDNSHP`)
7	`MRNM`	Mate Reference NaMe (‘=’ if same as `RNAME`)
8	`MPOS`	1-Based leftmost Mate POSition
9	`ISIZE`	Inferred Insert SIZE
10	`SEQ`	Query SEQuence on the same strand as the reference
11	`QUAL`	Query QUALity (ASCII-33=Phred base quality)

SAM format – Optional fields



Tag	Type	Description
X?	?	Reserved fields for end users (together with Y? and Z?)
AM	i	The smallest template-independent mapping quality of fragments in the rest
AS	i	Alignment score generated by aligner
BQ	Z	Offset to base alignment quality (BAQ), of the same length as the read sequence. At the i-th read base, $\mathrm{BAQ}_i = Q_i - (\mathrm{BQ}_i - 64)$ where Q_i is the i-th base quality.
CM	i	Edit distance between the color sequence and the color reference (see also NM)
CQ	Z	Color read quality on the original strand of the read. Same encoding as QUAL; same length as CS.
CS	Z	Color read sequence on the original strand of the read. The primer base must be included.
E2	Z	The 2nd most likely base calls. Same encoding and same length as QUAL.
FI	i	The index of fragment in the template.
FS	Z	Fragment suffix.
LB	Z	Library. Value to be consistent with the header RG-LB tag if @RG is present.
H0	i	Number of perfect hits
H1	i	Number of 1-difference hits (see also NM)
H2	i	Number of 2-difference hits
HI	i	Query hit index, indicating the alignment record is the i-th one stored in SAM
IH	i	Number of stored alignments in SAM that contains the query in the current record
MD	Z	String for mismatching positions. *Regex*: `[0-9]+(([ACGTN]|\^[ACGTN]+)[0-9]+)*`[1]
MQ	i	Mapping quality of the mate/next fragment
NH	i	Number of reported alignments that contains the query in the current record
NM	i	Edit distance to the reference, including ambiguous bases but excluding clipping
OQ	Z	Original base quality (usually before recalibration). Same encoding as QUAL.
OP	i	Original mapping position (usually before realignment)
OC	Z	Original CIGAR (usually before realignment)
PG	Z	Program. Value matches the header PG-ID tag if @PG is present.
PQ	i	Phred likelihood of the template, conditional on both the mapping being correct
PU	Z	Platform unit. Value to be consistent with the header RG-PU tag if @RG is present.
Q2	Z	Phred quality of the mate/next fragment. Same encoding as QUAL.
R2	Z	Sequence of the mate/next fragment in the template.
RG	Z	Read group. Value matches the header RG-ID tag if @RG is present in the header.
SM	i	Template-independent mapping quality
TC	i	The number of fragments in the template.
U2	Z	Phred probility of the 2nd call being wrong conditional on the best being wrong. The same encoding as QUAL.
UQ	i	Phred likelihood of the fragment, conditional on the mapping being correct

SAM output



amaxwell@pinfish:~/ecoli/illum/bwamem/i4m1_1/seqAssist1/result

```
[amaxwell@pinfish result]$ head -10 aln.sam
@SQ     SN:gi|49175990|ref|NC_000913.2| LN:4639675
HWI-EAS397:8:1:1067:18713#CTTGTA        16      gi|49175990|ref|NC_000913.2|    2909788 35      13S36M  *       0       0       TACAATAACCCCCCGCCCCTGGGTAATAAAGCCGACAATCTCATCTCCA       BBBBBBBBBBBBBBB^aa\^``\V_dKWdfad
affccffcY]adcL]^Y       NM:i:1  AS:i:31 XS:i:0
HWI-EAS397:8:1:1070:11813#CTTGTA        16      gi|49175990|ref|NC_000913.2|    4516890 24      49M     *       0       0       CGTCGGTGCTAAAGCAGGTCAGCGCTGGCTGTTTTACGCGTATGACAGG       BaYa[\KaaOGVHLZLIXH`]VaJR]V]V_J]
Qca]\aaS[J_aacccc       NM:i:2  AS:i:39 XS:i:30
HWI-EAS397:8:1:1070:11813#CTTGTA        256     gi|49175990|ref|NC_000913.2|    278711  0       30M19H  *       0       0       *       *       NM:i:0  AS:i:30
HWI-EAS397:8:1:1070:11813#CTTGTA        256     gi|49175990|ref|NC_000913.2|    290182  0       30M19H  *       0       0       *       *       NM:i:0  AS:i:30
HWI-EAS397:8:1:1070:11813#CTTGTA        272     gi|49175990|ref|NC_000913.2|    1049415 0       19H30M  *       0       0       *       *       NM:i:0  AS:i:30
HWI-EAS397:8:1:1121:19907#CTTGTA        0       gi|49175990|ref|NC_000913.2|    953460  60      49M     *       0       0       AGAGAGAAGCAAATGCCGCCAACCAGTTTTGCCATGCCGAAGGGCATTG       SIaJaL^ZT`[da^WcacfK^acSacfffff[
cY^dcdd^f]\]cffff       NM:i:0  AS:i:49 XS:i:0
HWI-EAS397:8:1:1138:20219#CTTGTA        4       *       0       0       *       *       0       0       TAAAAAAAGGCCATAACCTGCCGCTTTGTATAATAAAAAAGGGCCCGCG       Yd[cK^___Wdc\ddada]df[fccff_f_ffffcfdcdBBBBBBBBB        A
S:i:0   XS:i:0
HWI-EAS397:8:1:1149:21173#GTTGTA        4       *       0       0       *       *       0       0       GTTCGTTTTTTTTGTACAGTGCCAACATCTGCTCCCGGCCAATGGTGGC       Scc_ZccccaVZR^[W_K__M_\^VSQ]\\\aS^aVaXL]WY^^IaaaB        A
S:i:0   XS:i:0
HWI-EAS397:8:1:1152:20684#CTTGTA        4       *       0       0       *       *       0       0       GCTGATGCGTACGTGAAAGGGCGGGACACCTTGGATGTATGGTTTGAGT       aQ^aZ^MOaXOX_V\_cc_BBBBBBBBBBBBBBBBBBBBBBBBBBBBB        A
S:i:0   XS:i:0
[amaxwell@pinfish result]$
```

Contents

- **Alignment algorithms for short-reads**
  - Background – Blast (why can't we use it?)
  - Adapting hashed seed-extend algorithms to work with shorter reads
  - Indel detection
  - Suffix/Prefix Tries
  - **Other alignment considerations**
  - Typical alignment pipeline
  - New methods of SNP calling

Other alignment considerations

- Indel detection
- Effect of paired-end alignments
- Using base quality to inform alignments
- PCR duplicates
- Methylation experiments – bisulfite treated reads
- Multi-mapping reads
- Aligning spliced-reads from RNA-seq experiments
- Local realignment to improve SNP/Indel detection
- Platform specific errors
- Unmapped reads

Indel detection

Spaced seed with weight 9bp and no mismatches:

ACTCCCATTGTCATCGTACTTGGGATCGTAACA Reference sequence

CCATTGTCATGTACTTGGGATCGT Read containing a deletion

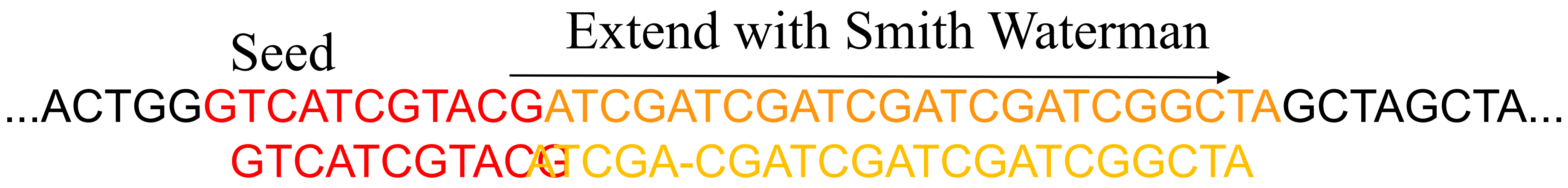
CCATTGTCATCGTACAT

CCXXTGXXATXXACXXG Seed not matched due to frame shift caused by gap

No seed match. No alignment!

Indel detection

Reference sequence:



```
                  Seed        Extend with Smith Waterman
                             ------------------------------------->
...ACTGGGTCATCGTACGATCGATCGATCGATCGATCGGCTAGCTAGCTA...
        GTCATCGTACGATCGA-CGATCGATCGATCGGCTA
```

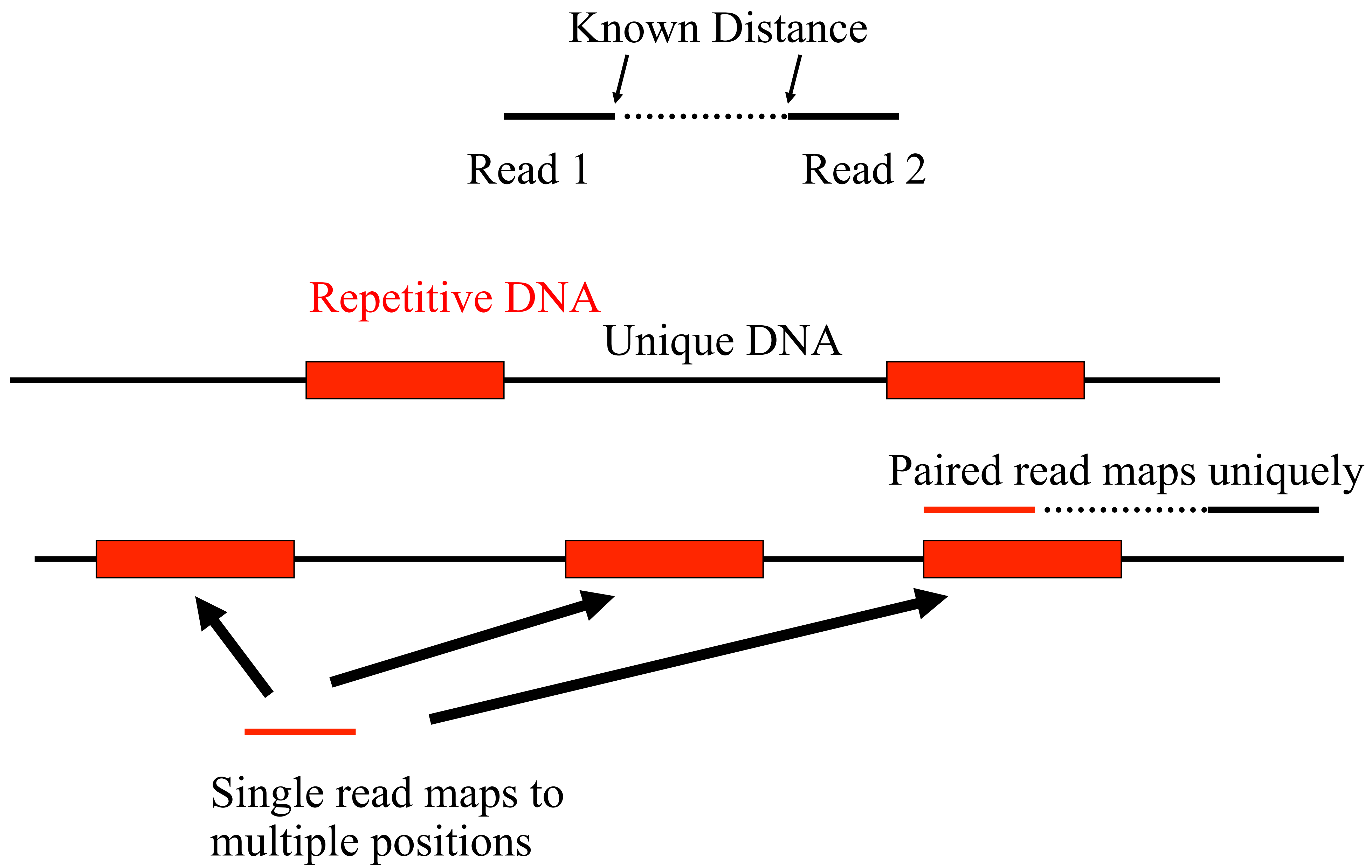
Most alignment programs can only detect gaps in Smith-Waterman phase
once a seed has been identified. Some algorithms (e.g. Bowtie) do not allow gaps at this stage to improve speed

This reduces sensitivity especially with multiple insertions in a small region

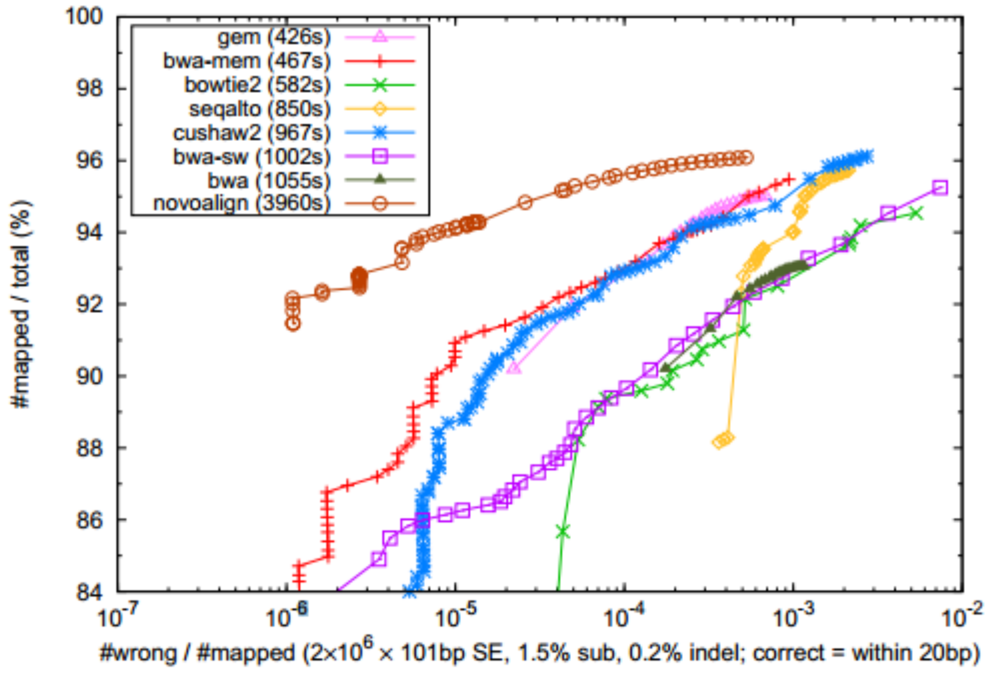
Indel detection

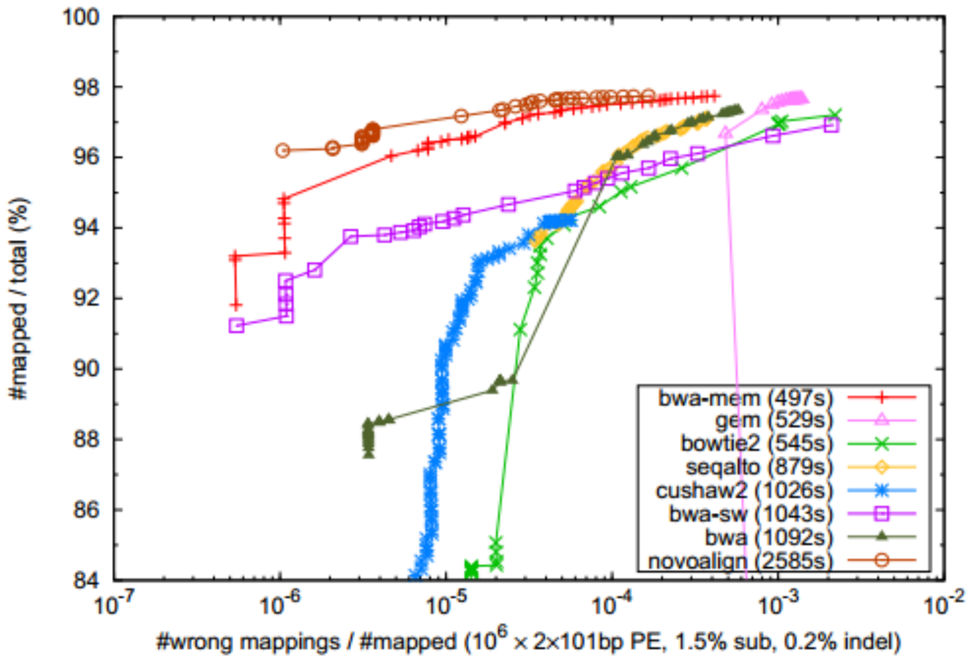
- Some algorithms do allow gaps within seed
  - Indel seeds for homology search *Bioinformatics (2006) 22(14): e341-e349 doi:10.1093/bioinformatics/btl263*
  - Weese D, Emde AK, Rausch T, et al. RazerS–fast read mapping with sensitivity control. Genome Res 2009;19:1646–54
  - Rumble SM, Lacroute P, Dalca AV, et al. SHRiMP: accurate mapping of short color-space reads. PLoS Comput Biol 2009;5:e1000386
- Use of multiple seeds
  - Especially useful for longer reads (>50bp)
  - Li R, Li Y, Kristiansen K, et al. SOAP: short oligonucleotide alignment program. Bioinformatics 2008;24:713–4
  - Jiang H, Wong WH. SeqMap: mapping massive amount of oligonucleotides to the genome. Bioinformatics 2008;24: 2395–6

Paired-end reads are important


Known Distance
Read 1
Read 2
Repetitive DNA
Unique DNA
Paired read maps uniquely
Single read maps to multiple positions

Effect of paired-end alignments



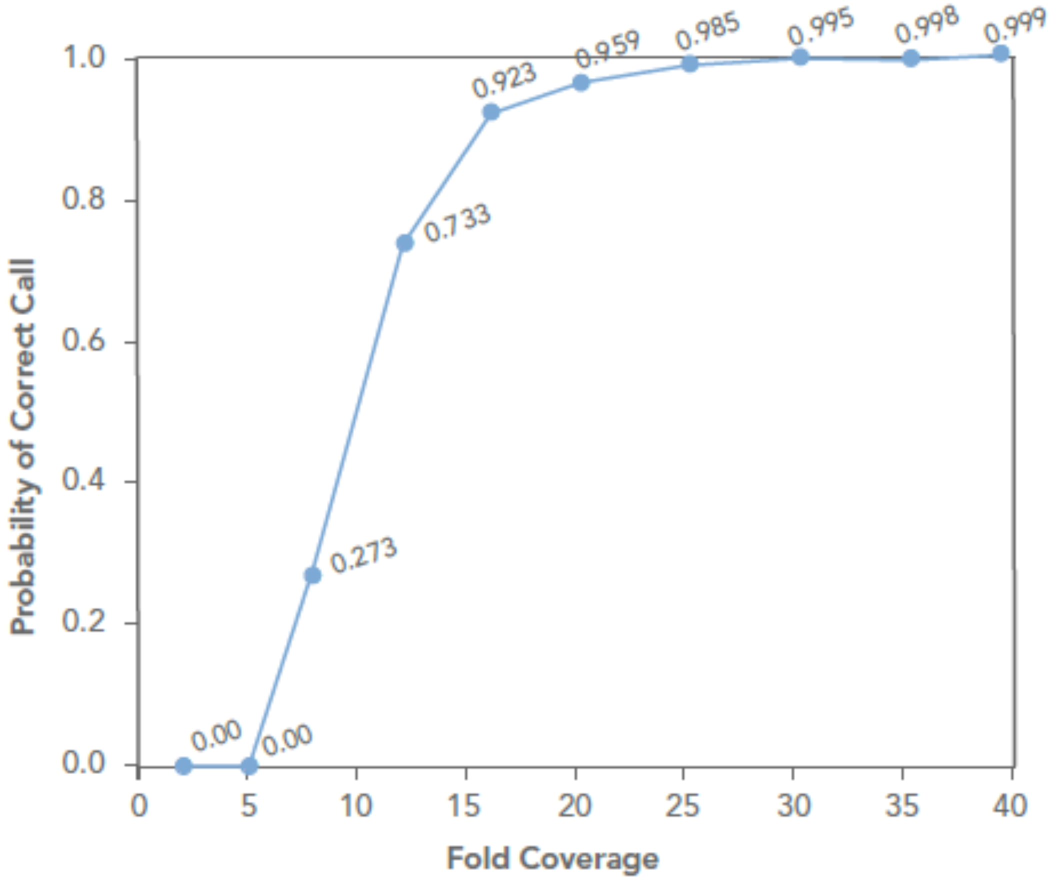


BWA-MEM

http://arxiv.org/pdf/1303.3997v2.pdf

Effect of coverage on SNP call accuracy

- Depends on ploidy
- Bacterial genomes can get away with 10-20x
- For human genomes and other diploids 30x
- Poly-ploids (e.g wheat) may need much higher coverage



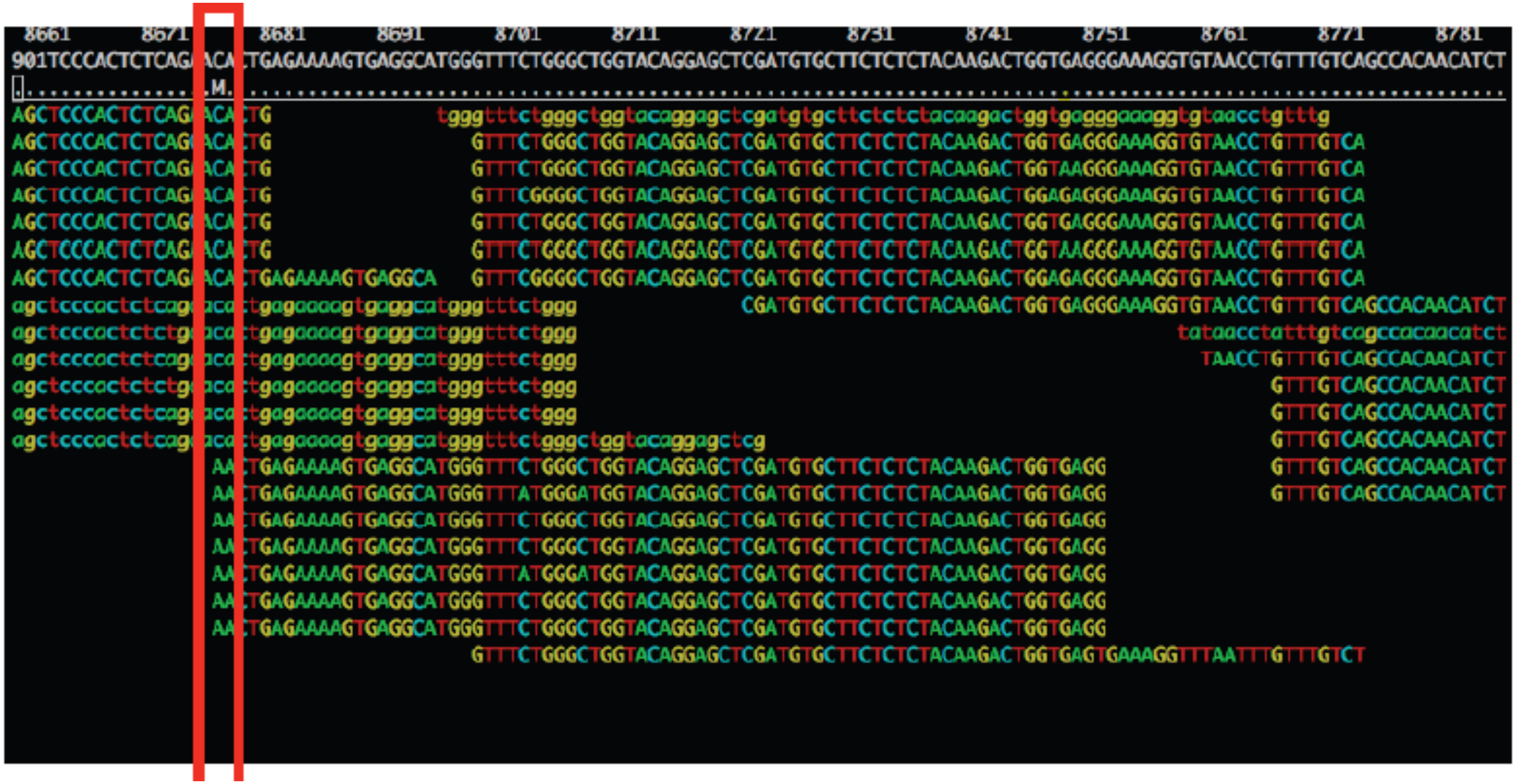
Source – Illumina Tech Note
Human diploid sample

PCR duplicates

- **2nd generation sequencers are not single-molecule sequencers**
  - All have at least one PCR amplification step
  - Can result in duplicate DNA fragments
  - This can bias SNP calls or introduce false SNPs

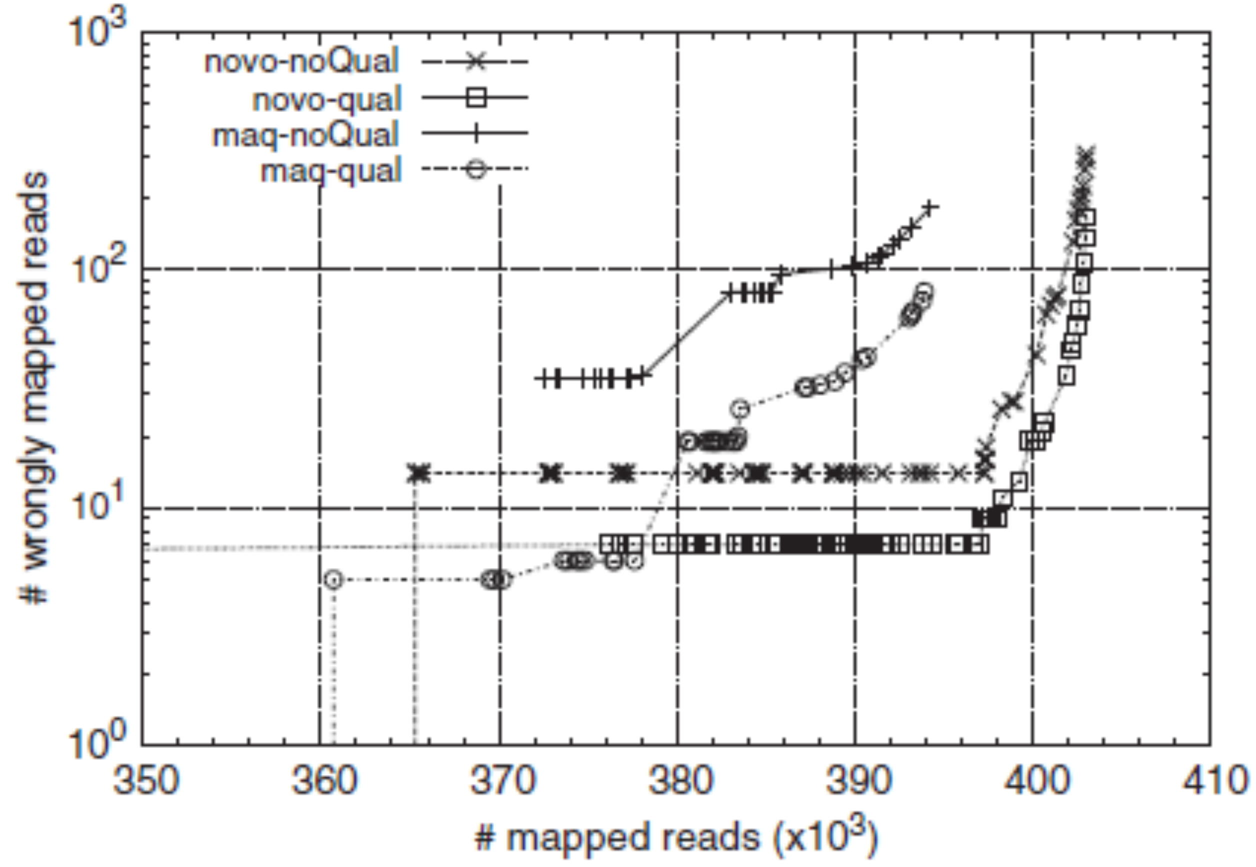
- **Generally duplicates only make up a small fraction of the results**
  - Good libraries have < 2-3% of duplicates
  - SAMtools and Picard can identify and remove these when aligned against a reference genome
  - Debatable whether do this for RNA-seq and ChIP-seq
    - Depends on the complexity of the sample

PCR duplicates



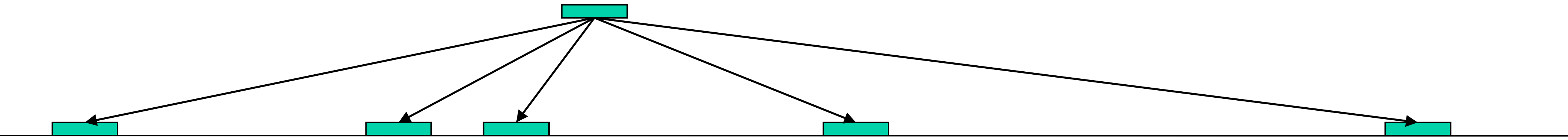
```
  8661      8671      8681      8691      8701      8711      8721      8731      8741      8751      8761      8771      8781
901TCCCACTCTCAG ACA TGAGAAAAGTGAGGCATGGGTTTCTGGGCTGGTACAGGAGCTCGATGTGCTTCTCTCTACAAGACTGGTGAGGGAAAGGTGTAACCTGTTTGTCAGCCACAACATCT
.................M.............................................................................................................
AGCTCCCACTCTCAG ACA TG              tgggtttctgggctggtacaggagctcgatgtgcttctctctacaagactggtgagggaaaggtgtaacctgtttg
AGCTCCCACTCTCAG ACA TG                 GTTTCTGGGCTGGTACAGGAGCTCGATGTGCTTCTCTCTACAAGACTGGTGAGGGAAAGGTGTAACCTGTTTGTCA
AGCTCCCACTCTCAG ACA TG                 GTTTCTGGGCTGGTACAGGAGCTCGATGTGCTTCTCTCTACAAGACTGGTAAGGGAAAGGTGTAACCTGTTTGTCA
AGCTCCCACTCTCAG ACA TG                 GTTTCGGGGCTGGTACAGGAGCTCGATGTGCTTCTCTCTACAAGACTGGAGAGGGAAAGGTGTAACCTGTTTGTCA
AGCTCCCACTCTCAG ACA TG                 GTTTCTGGGCTGGTACAGGAGCTCGATGTGCTTCTCTCTACAAGACTGGTGAGGGAAAGGTGTAACCTGTTTGTCA
AGCTCCCACTCTCAG ACA TG                 GTTTCTGGGCTGGTACAGGAGCTCGATGTGCTTCTCTCTACAAGACTGGTAAGGGAAAGGTGTAACCTGTTTGTCA
AGCTCCCACTCTCAG ACA TGAGAAAAGTGAGGCA   GTTTCGGGGCTGGTACAGGAGCTCGATGTGCTTCTCTCTACAAGACTGGAGAGGGAAAGGTGTAACCTGTTTGTCA
agctcccactctcag aca tgagaaaagtgaggcatgggtttctggg              CGATGTGCTTCTCTCTACAAGACTGGTGAGGGAAAGGTGTAACCTGTTTGTCAGCCACAACATCT
agctcccactctctg aca tgagaaaagtgaggcatgggtttctggg                                                   tataacctatttgtcagccacaacatct
agctcccactctcag aca tgagaaaagtgaggcatgggtttctggg                                                     TAACCTGTTTGTCAGCCACAACATCT
agctcccactctctg aca tgagaaaagtgaggcatgggtttctggg                                                           GTTTGTCAGCCACAACATCT
agctcccactctcag aca tgagaaaagtgaggcatgggtttctggg                                                           GTTTGTCAGCCACAACATCT
agctcccactctcag aca tgagaaaagtgaggcatgggtttctgggctggtacaggagctcg                                           GTTTGTCAGCCACAACATCT
                 AA TGAGAAAAGTGAGGCATGGGTTTCTGGGCTGGTACAGGAGCTCGATGTGCTTCTCTCTACAAGACTGGTGAGG               GTTTGTCAGCCACAACATCT
                 AA TGAGAAAAGTGAGGCATGGGTTTATGGGATGGTACAGGAGCTCGATGTGCTTCTCTCTACAAGACTGGTGAGG               GTTTGTCAGCCACAACATCT
                 AA TGAGAAAAGTGAGGCATGGGTTTCTGGGCTGGTACAGGAGCTCGATGTGCTTCTCTCTACAAGACTGGTGAGG
                 AA TGAGAAAAGTGAGGCATGGGTTTCTGGGCTGGTACAGGAGCTCGATGTGCTTCTCTCTACAAGACTGGTGAGG
                 AA TGAGAAAAGTGAGGCATGGGTTTATGGGATGGTACAGGAGCTCGATGTGCTTCTCTCTACAAGACTGGTGAGG
                 AA TGAGAAAAGTGAGGCATGGGTTTCTGGGCTGGTACAGGAGCTCGATGTGCTTCTCTCTACAAGACTGGTGAGG
                 AA TGAGAAAAGTGAGGCATGGGTTTCTGGGCTGGTACAGGAGCTCGATGTGCTTCTCTCTACAAGACTGGTGAGG
                                       GTTTCTGGGCTGGTACAGGAGCTCGATGTGCTTCTCTCTACAAGACTGGTGAGTGAAAGGTTTAATTTGTTTGTCT
```

Base quality impacts on read mapping



Heng Li & Nils Homer. Sequence alignment algorithms for next-generation sequencing. Briefings in Bioinformatics. Vol 11. No 5. 473 483, 2010

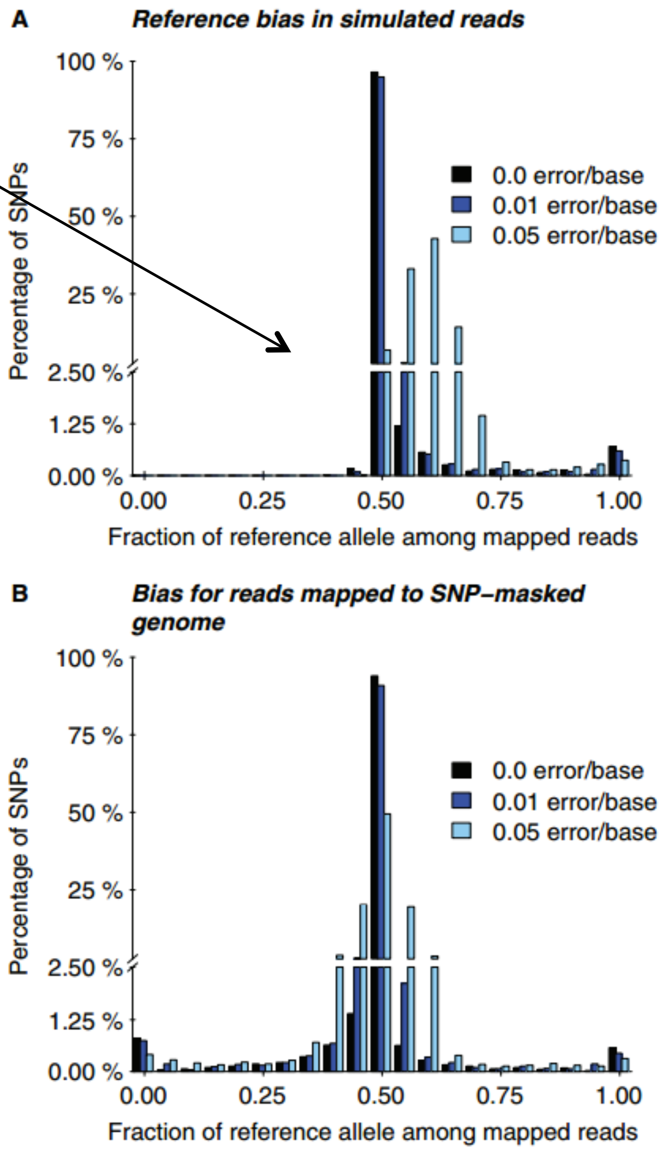
Multiple mapping reads



- A single read may occur more than once in the reference genome.
- Could be due to:
    - Paralogs (duplicated genes).
    - Transcripts which share exons.
    - Mutations in genotype relative to the reference.
    - Transposons and other common repetitive sequences
- Some aligners automatically assign a multi-mapping read to one of the locations at random (e.g. Tophat)
- Aligners may allow you to chose how these are dealt with – others may not

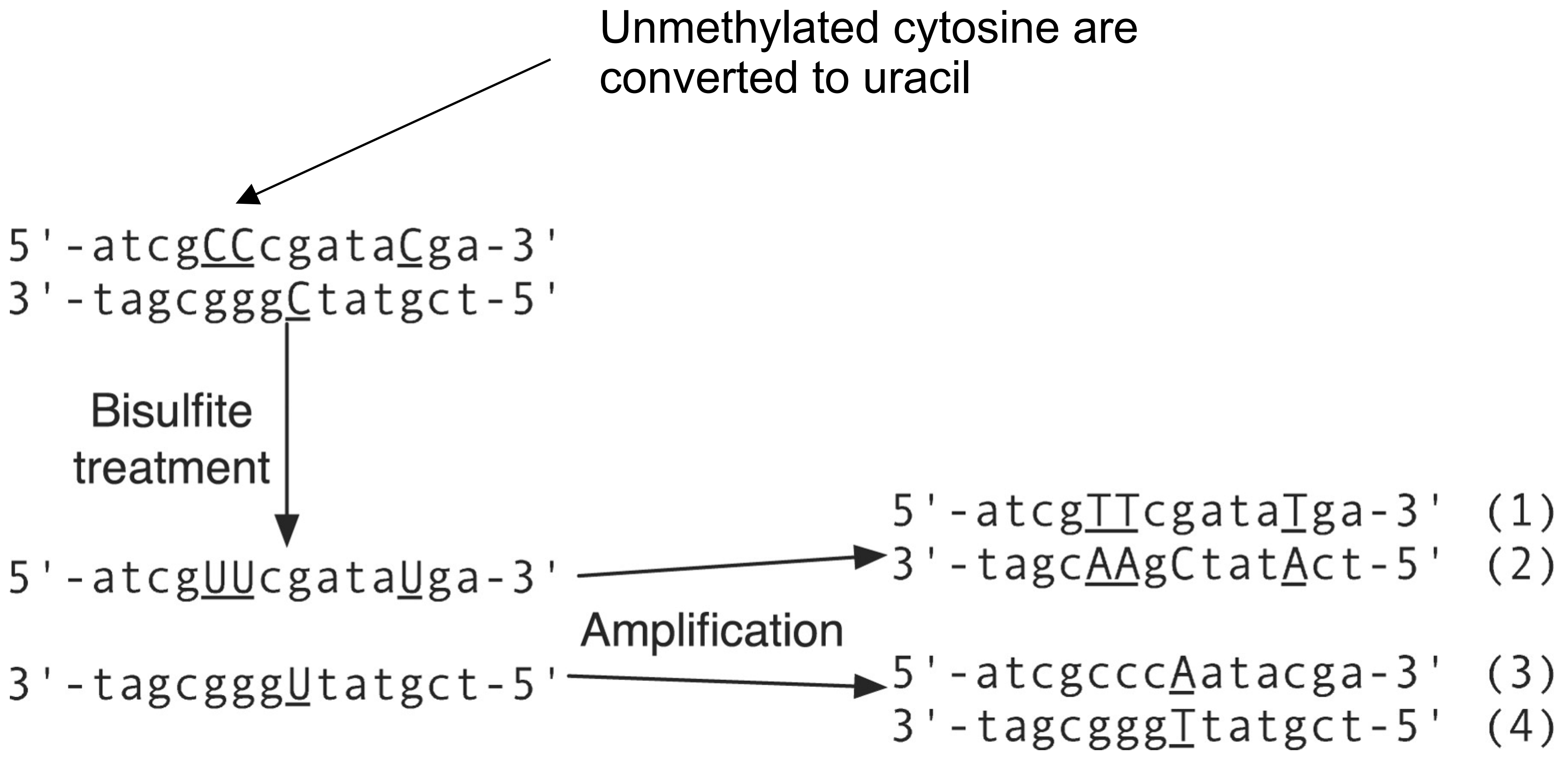
Allelic bias when SNP calling

Missing alternate allele



http://bioinformatics.oxfordjournals.org/content/25/24/3207.full.pdf

Methylation experiments



Methylation experiments

• Directly aligning reads against a reference will fail due to excessive mismatches in non-methylated regions

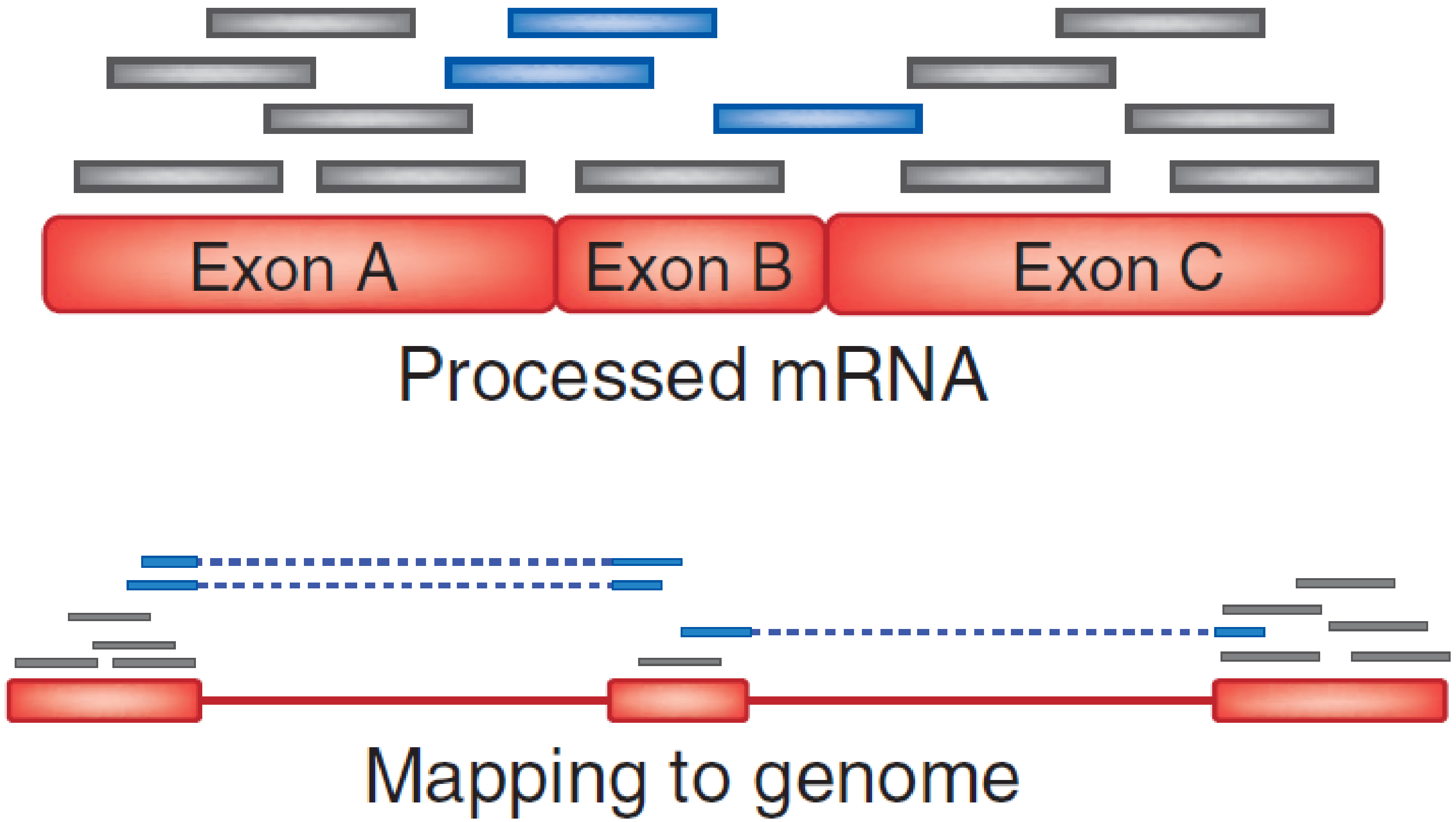
• Most packages deal with this by creating 2 reference sequences
  - One has all Cs converted to Ts
  - One has all Gs converted to As

• Convert Cs to Ts in all reads aligned against C->T reference
• Convert Gs to As in all reads aligned against G->A reference

• If there are no mutations or sequencing errors the reads will always map to one of the two references

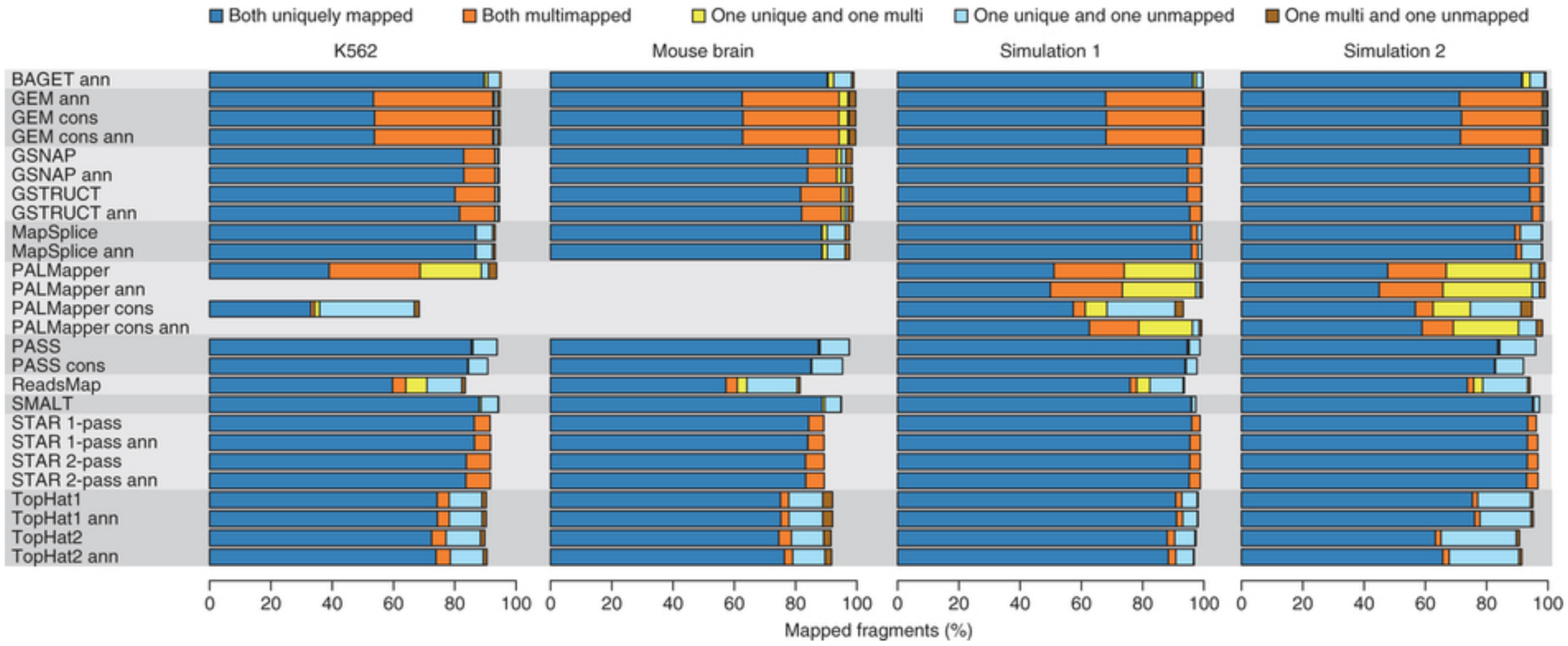
http://www.biomedcentral.com/1471-2164/14/774

Spliced-read mapping



- Need packages which can account for splice variants
- Examples: TopHat, STAR, GMAP, MapSplice

Spliced-read mapper evaluation



http://www.nature.com/nmeth/journal/v10/n12/full/nmeth.2722.html

Local realignment to improve SNP/Indel detection

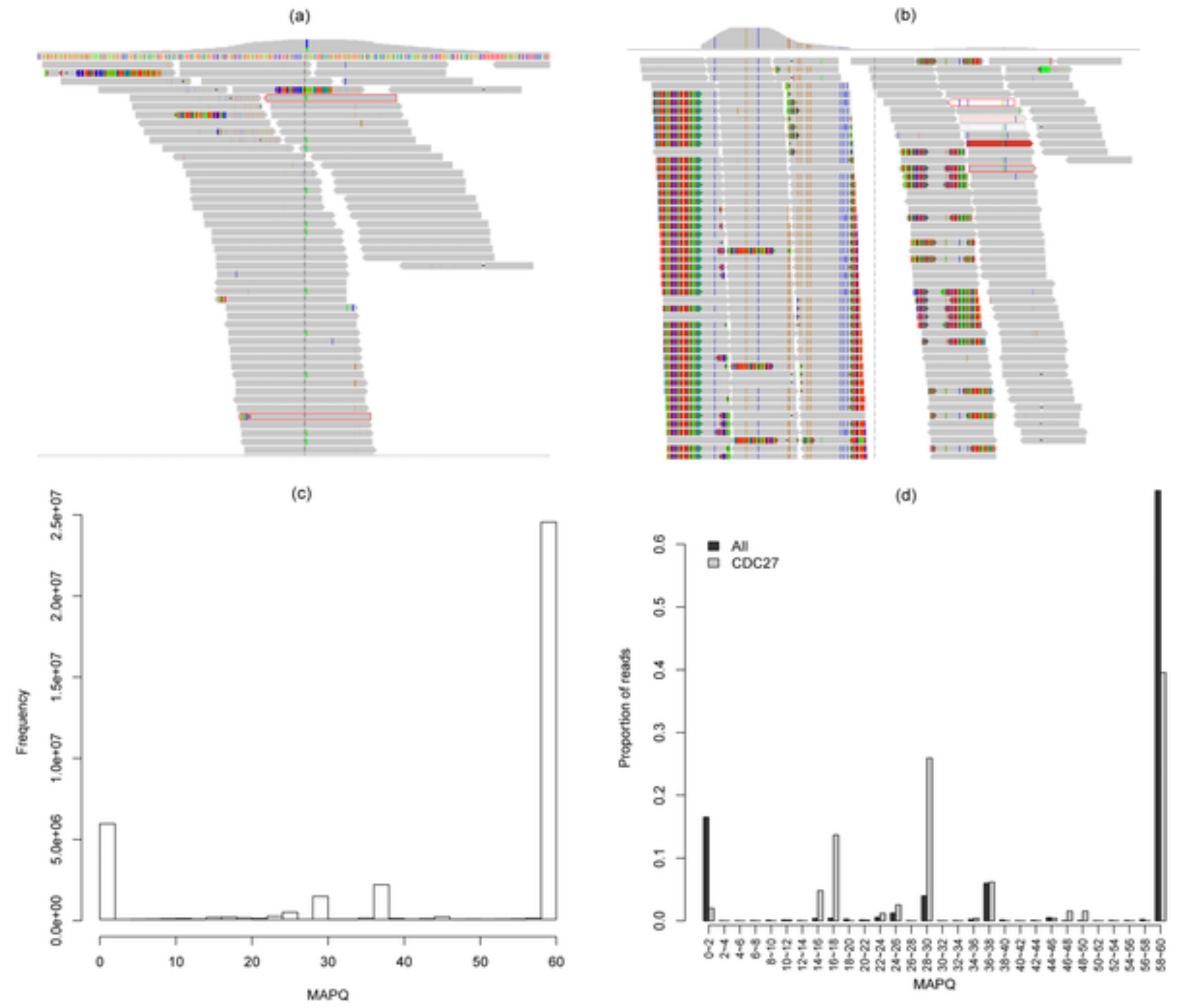
- Read aligners map each read (or read pair) independently of all other reads
- Around indels and other variants it can be helpful to make use of other metrics

  e.g. Global median coverage for multi-mapping reads
- Tools such as GATK, SAMtools, Pindel and Breakdancer realign reads in the vicinity of variants to improve calls

http://www.broadinstitute.org/gsa/wiki/index.php/The_Genome_Analysis_Toolkit

Chen, K. BreakDancer: an algorithm for high-resolution mapping of genomic structural variation *Nature Methods* 6, 677 - 681 (2009)
Li H.*, Handsaker B.*, Wysoker A., Fennell T., Ruan J., Homer N., Marth G., Abecasis G., Durbin R. and 1000 Genome Project Data Processing Subgroup (2009) The Sequence alignment/map (SAM) format and SAMtools. Bioinformatics, 25, 2078-9

Figure 6. A visual examination of a spurious gene (CDC27).



http://www.plosone.org/article/info:doi/10.1371/journal.pone.0038470

All platforms have errors and artefacts

Illumina

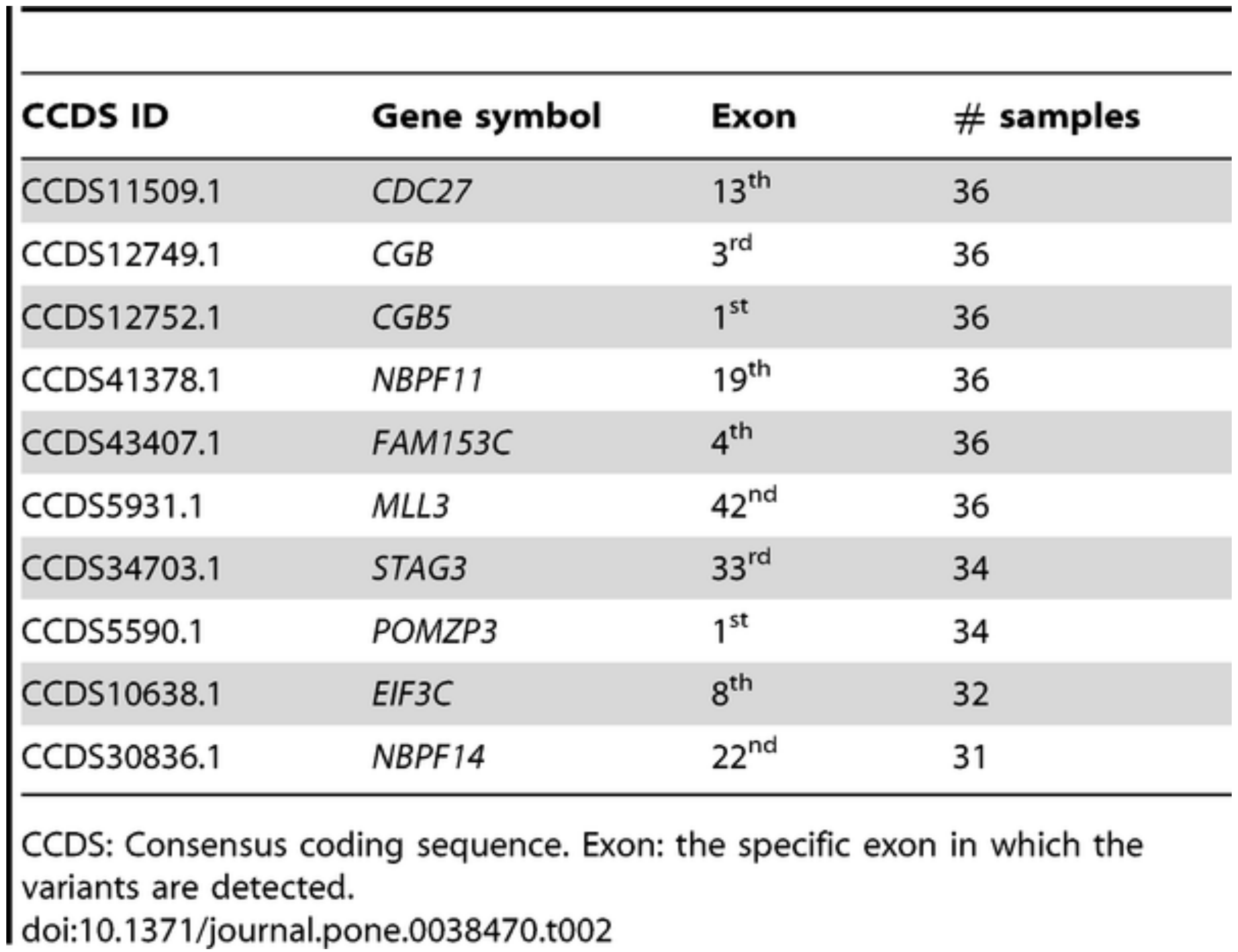
PacBio

Roche 454

Ion Torrent

1. Removal of low quality bases
2. Removal of adaptor sequences
3. Platform specific artefacts (e.g homopolymers)

Table 2. Spurious genes having mutations detected in 30 samples.

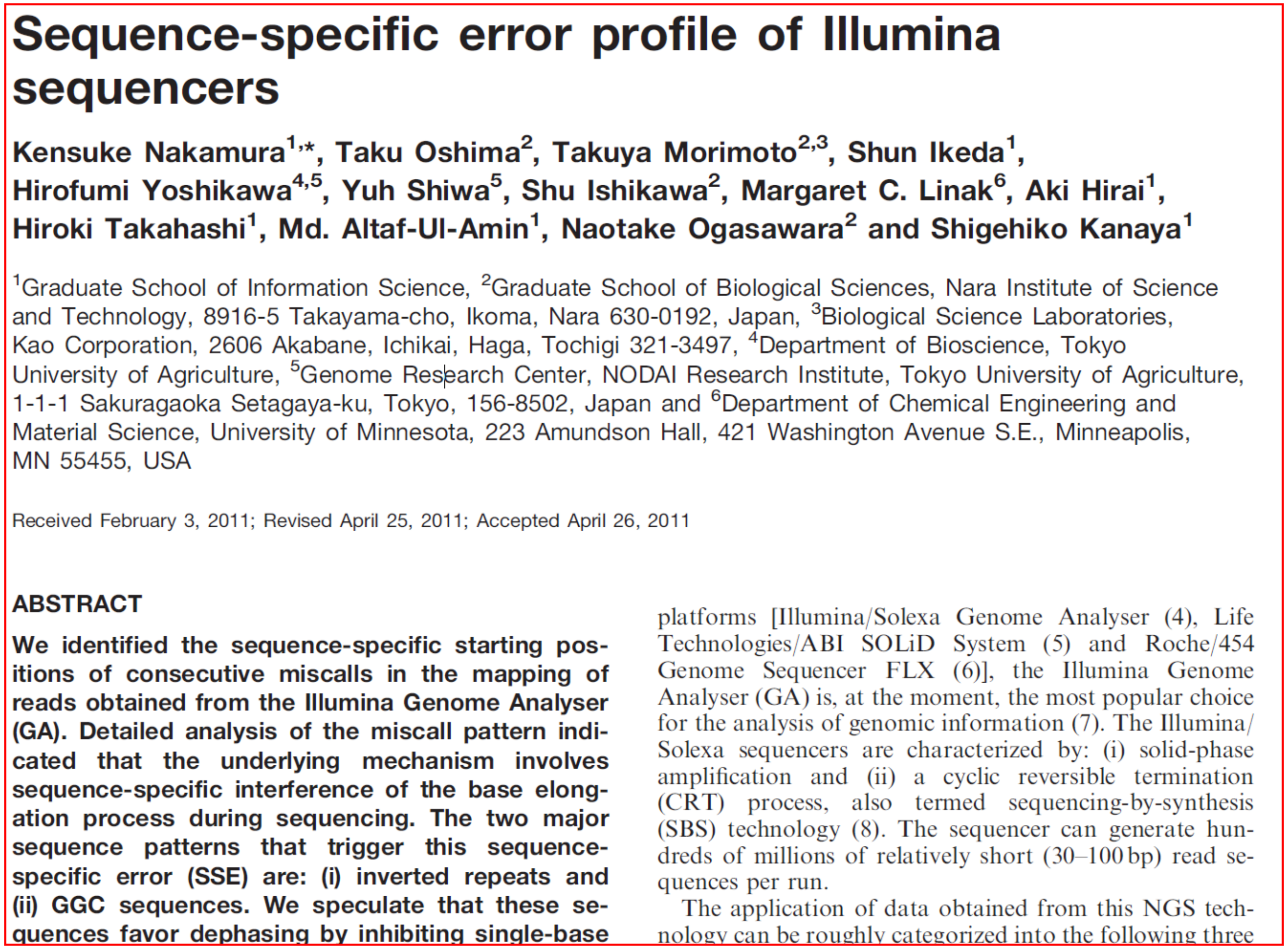


CCDS ID	Gene symbol	Exon	# samples
CCDS11509.1	*CDC27*	13th	36
CCDS12749.1	*CGB*	3rd	36
CCDS12752.1	*CGB5*	1st	36
CCDS41378.1	*NBPF11*	19th	36
CCDS43407.1	*FAM153C*	4th	36
CCDS5931.1	*MLL3*	42nd	36
CCDS34703.1	*STAG3*	33rd	34
CCDS5590.1	*POMZP3*	1st	34
CCDS10638.1	*EIF3C*	8th	32
CCDS30836.1	*NBPF14*	22nd	31

CCDS: Consensus coding sequence. Exon: the specific exon in which the variants are detected.

doi:10.1371/journal.pone.0038470.t002

Illumina artefacts



Sequence-specific error profile of Illumina sequencers

Kensuke Nakamura[1,*], Taku Oshima[2], Takuya Morimoto[2,3], Shun Ikeda[1], Hirofumi Yoshikawa[4,5], Yuh Shiwa[5], Shu Ishikawa[2], Margaret C. Linak[6], Aki Hirai[1], Hiroki Takahashi[1], Md. Altaf-Ul-Amin[1], Naotake Ogasawara[2] and Shigehiko Kanaya[1]

[1]Graduate School of Information Science, [2]Graduate School of Biological Sciences, Nara Institute of Science and Technology, 8916-5 Takayama-cho, Ikoma, Nara 630-0192, Japan, [3]Biological Science Laboratories, Kao Corporation, 2606 Akabane, Ichikai, Haga, Tochigi 321-3497, [4]Department of Bioscience, Tokyo University of Agriculture, [5]Genome Research Center, NODAI Research Institute, Tokyo University of Agriculture, 1-1-1 Sakuragaoka Setagaya-ku, Tokyo, 156-8502, Japan and [6]Department of Chemical Engineering and Material Science, University of Minnesota, 223 Amundson Hall, 421 Washington Avenue S.E., Minneapolis, MN 55455, USA

Received February 3, 2011; Revised April 25, 2011; Accepted April 26, 2011

ABSTRACT

We identified the sequence-specific starting positions of consecutive miscalls in the mapping of reads obtained from the Illumina Genome Analyser (GA). Detailed analysis of the miscall pattern indicated that the underlying mechanism involves sequence-specific interference of the base elongation process during sequencing. The two major sequence patterns that trigger this sequence-specific error (SSE) are: (i) inverted repeats and (ii) GGC sequences. We speculate that these sequences favor dephasing by inhibiting single-base

platforms [Illumina/Solexa Genome Analyser (4), Life Technologies/ABI SOLiD System (5) and Roche/454 Genome Sequencer FLX (6)], the Illumina Genome Analyser (GA) is, at the moment, the most popular choice for the analysis of genomic information (7). The Illumina/Solexa sequencers are characterized by: (i) solid-phase amplification and (ii) a cyclic reversible termination (CRT) process, also termed sequencing-by-synthesis (SBS) technology (8). The sequencer can generate hundreds of millions of relatively short (30–100 bp) read sequences per run.

The application of data obtained from this NGS technology can be roughly categorized into the following three

Nakamura, K. et al. Sequence-specific error profile of Illumina sequencers
Nucl. Acids Res. (2011) May 16, 2011

Illumina artefacts

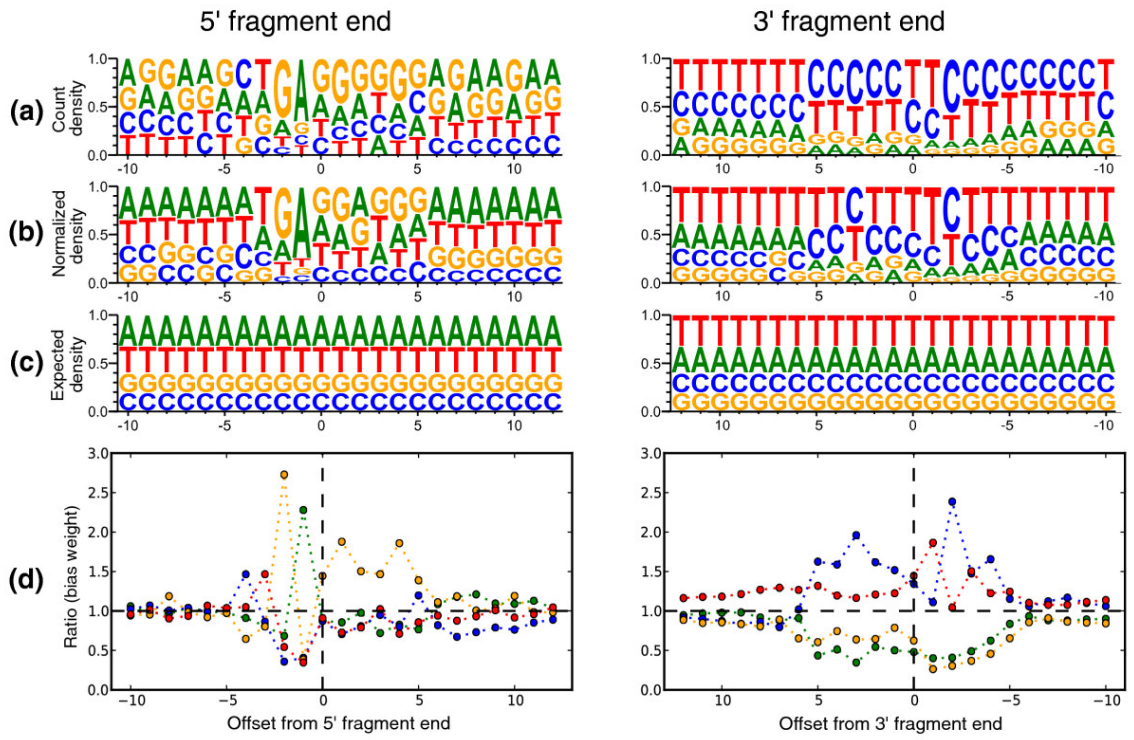
1. GC rich regions are under represented
    a. PCR
    b. Sequencing
2. Substitutions more common than insertions
3. GGC/GCC motif is associated with low quality and mismatches
4. Filtering low quality reads exacerbates low coverage of GC regions

Alignment software should ideally account for technology specific bias but generally does not

Your alignments are only as good as your library prep

- Even if all other artefacts are removed:
- If your library prep is biased, your alignments will also reflect this bias

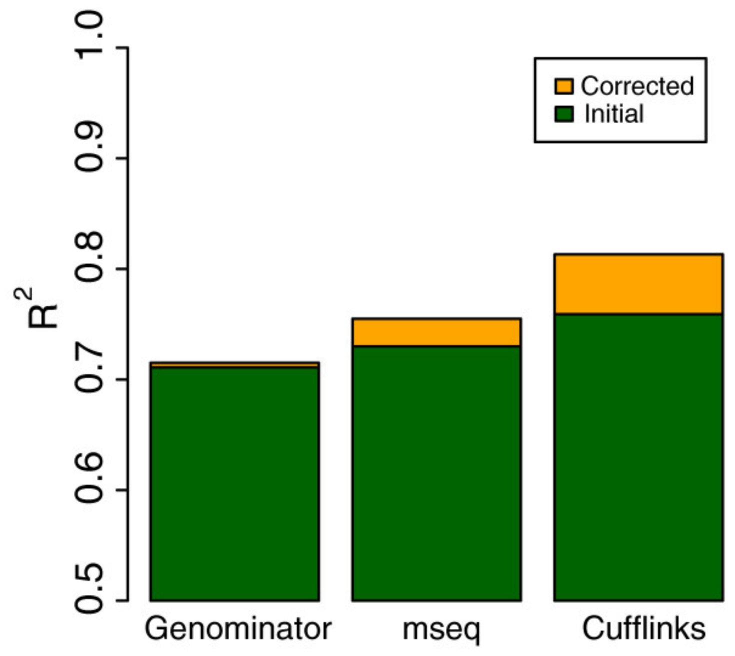
Tophat/Cufflinks aside

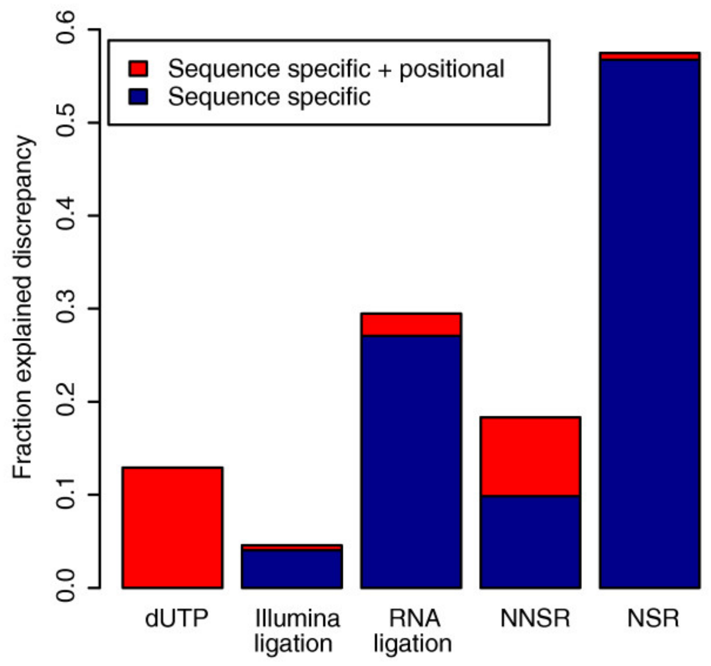


- Applies to random primed RNA-seq libraries
- Main potential biases:
  - Random hexamer priming biases
  - 5’ or 3’ ends of cDNA are likely to be mis-represented
  - Some packages correct for this (e.g. Tophat/Cufflinks)

http://genomebiology.com/2011/12/3/R22

Effect of bias correction



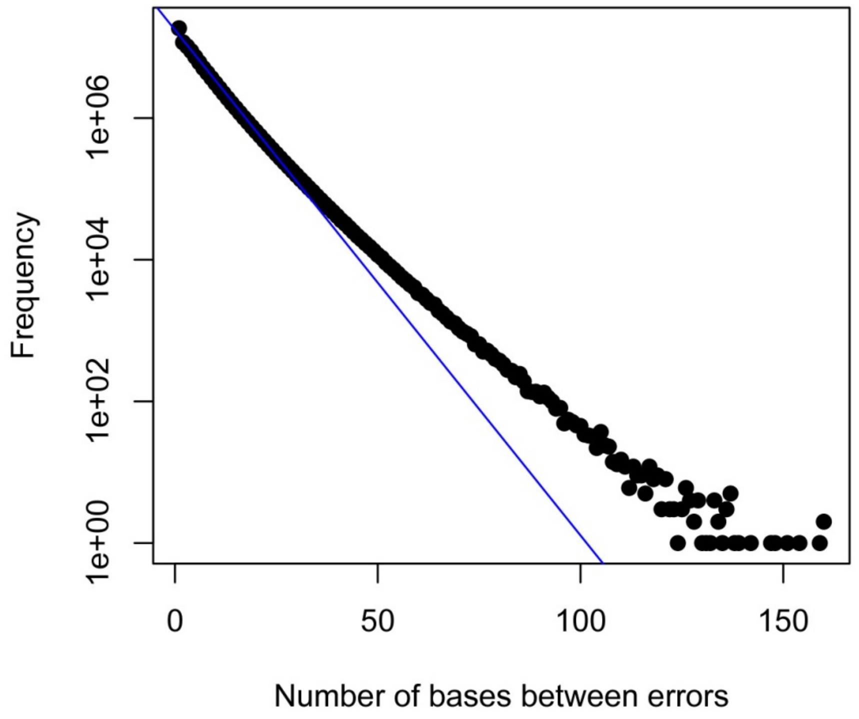


http://genomebiology.com/2011/12/3/R22

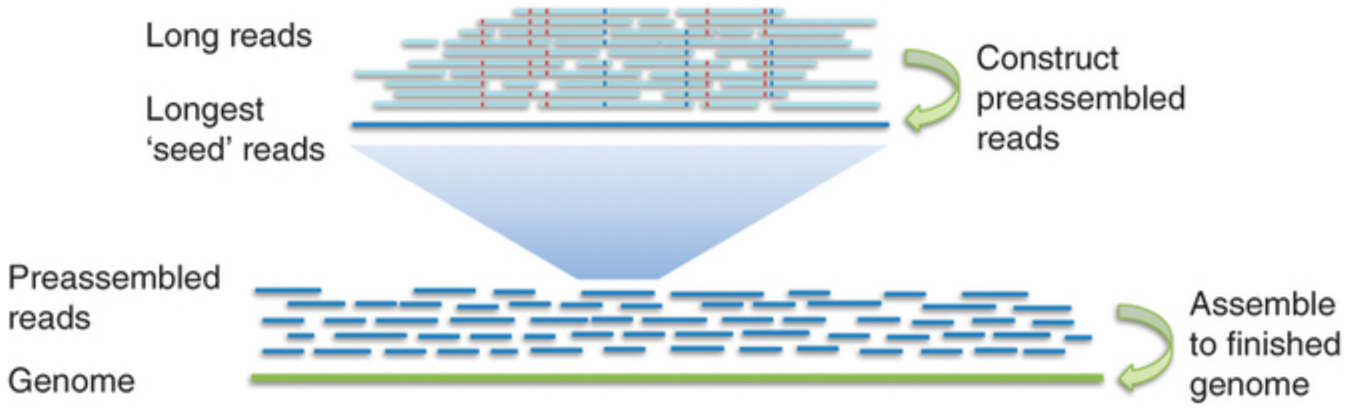
N.B. Out-dated version of Cufflinks used here

Pacific Biosciences alignment

- Median PacBio reads are 12kb with an single-read error rate of around 12-13%
- Most common distance between errors in a PacBio read is around 30bp
- Long length compensates and enables seeds to be located in many places to begin alignment
- However, this can be computationally costly
  - Need to balance number of seeds required to get a good alignment vs computational time
- PacBio developed BLASR to do this



Pacific Biosciences read correction via alignment

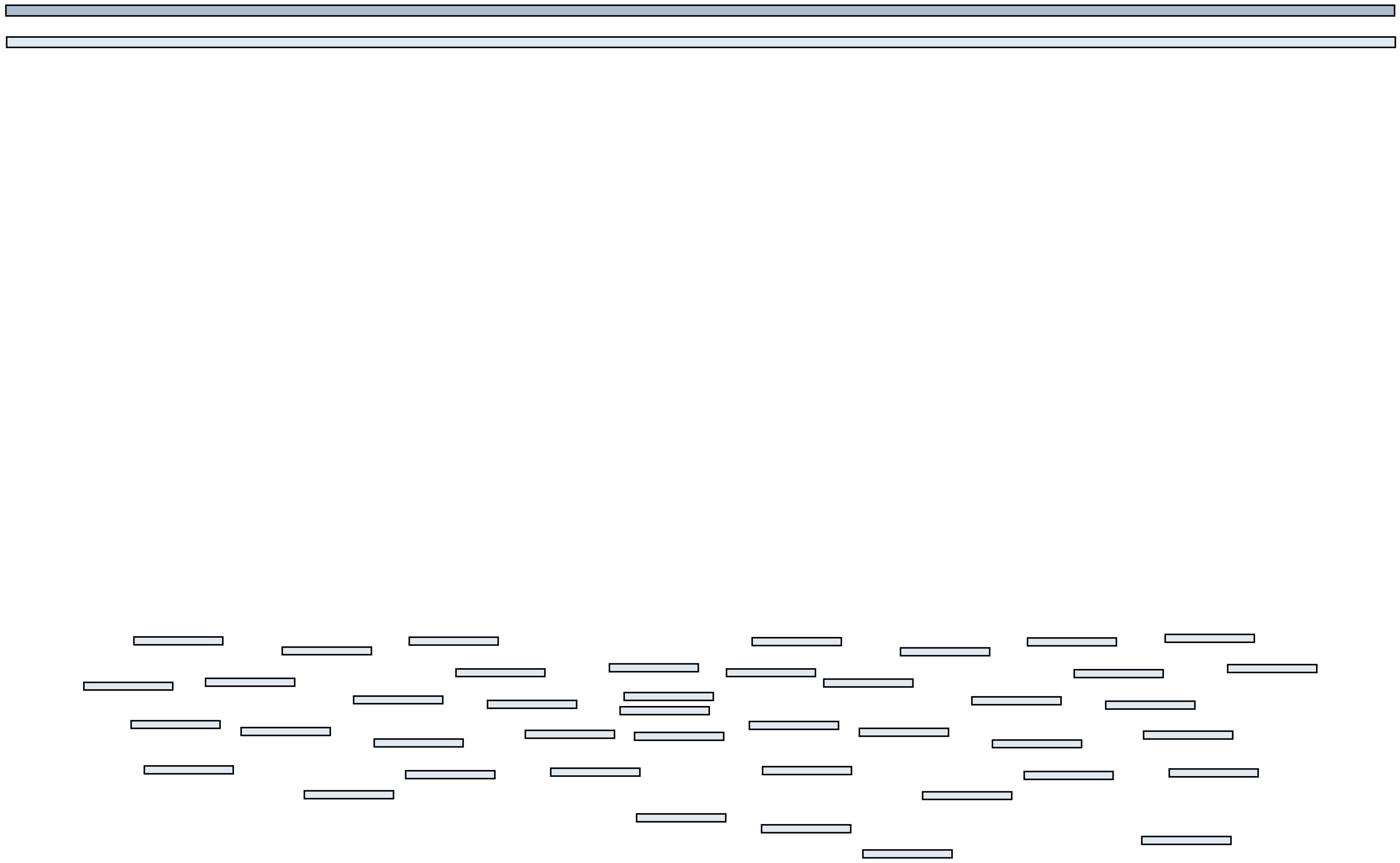


- Alignment can also be used to generate a consensus to reduce the number of errors
- Most errors in PacBio seem to be randomly distributed
- If we have enough coverage, we can correct these errors by aligning all the short reads to the longer reads and correct based on the *consensus*

- These can then be used for denovo assembly
- Hierarchical Genome Assembly Process (HGAP)

http://www.nature.com/nmeth/journal/v10/n6/full/nmeth.2474.html

Unmapped reads



Unmapped reads

- Can be the result of:
  - Sequencing errors (should be small fraction if quality filtering applied before mapping)
  - Contamination
  - Excessive matches to repeats
  - Highly divergent regions between samples
  - Novel genetic material not present in reference
  - Plasmids

- Should be assembled de-novo with paired-end information if possible
- Resulting contigs run through MegaBlast against NCBI NT to check species
- Check against RepBase to remove repetitive contigs
- Call ORFs
- Blast ORFs using BlastP against NCBI NR or Swissprot and Blast2GO
- Run through PFAM

Typical alignment pipeline

QC

- •Remove low quality bases
- •Remove reads containing adaptor sequences
- •Trim or remove reads containing Ns

Alignment

- •Generate reference or read index
- •Align reads to index
- •SAM output file

Post alignment

- •Sort SAM file and convert to BAM with SAMtools
- •Remove suspected PCR duplicates with SAMtools
- •Perform local realignment around indels using GATK
- •Supply BAM file to variant caller (e.g. Samtools mpileup)
- •Analyse variants (are they within genes, synonymous vs nonsynonymous changes etc)*
- •Locate missing genes/regulatory regions

Assemble unmapped reads

- •Assemble unmapped reads (e.g. using Velvet)
- •Call Open Reading Frames (ORFs)
- •Search for homologous genes (BLASTP), protein families (PFAM)
- •Identify novel genes

* http://bioinformatics.net.au/software.nesoni.shtml

Contents

- **Alignment algorithms for short-reads**
  - Background – Blast (why can't we use it?)
  - Adapting hashed seed-extend algorithms to work with shorter reads
  - Indel detection
  - Suffix/Prefix Tries
  - Other alignment considerations
  - Typical alignment pipeline
  - **Haplotype methods of variant calling**

Haploytype SNP calling

- FreeBayes (http://arxiv.org/pdf/1207.3907v2.pdf)
- GATK – Haplotype caller
  - Haplotype calling in polyploids

ACCTGTA Reference Genome

Assume a SNP at both 5’ A->T and 3’ A->G in a diploid

Do we have a heterozygous?

Allele 1: ACCTGTG

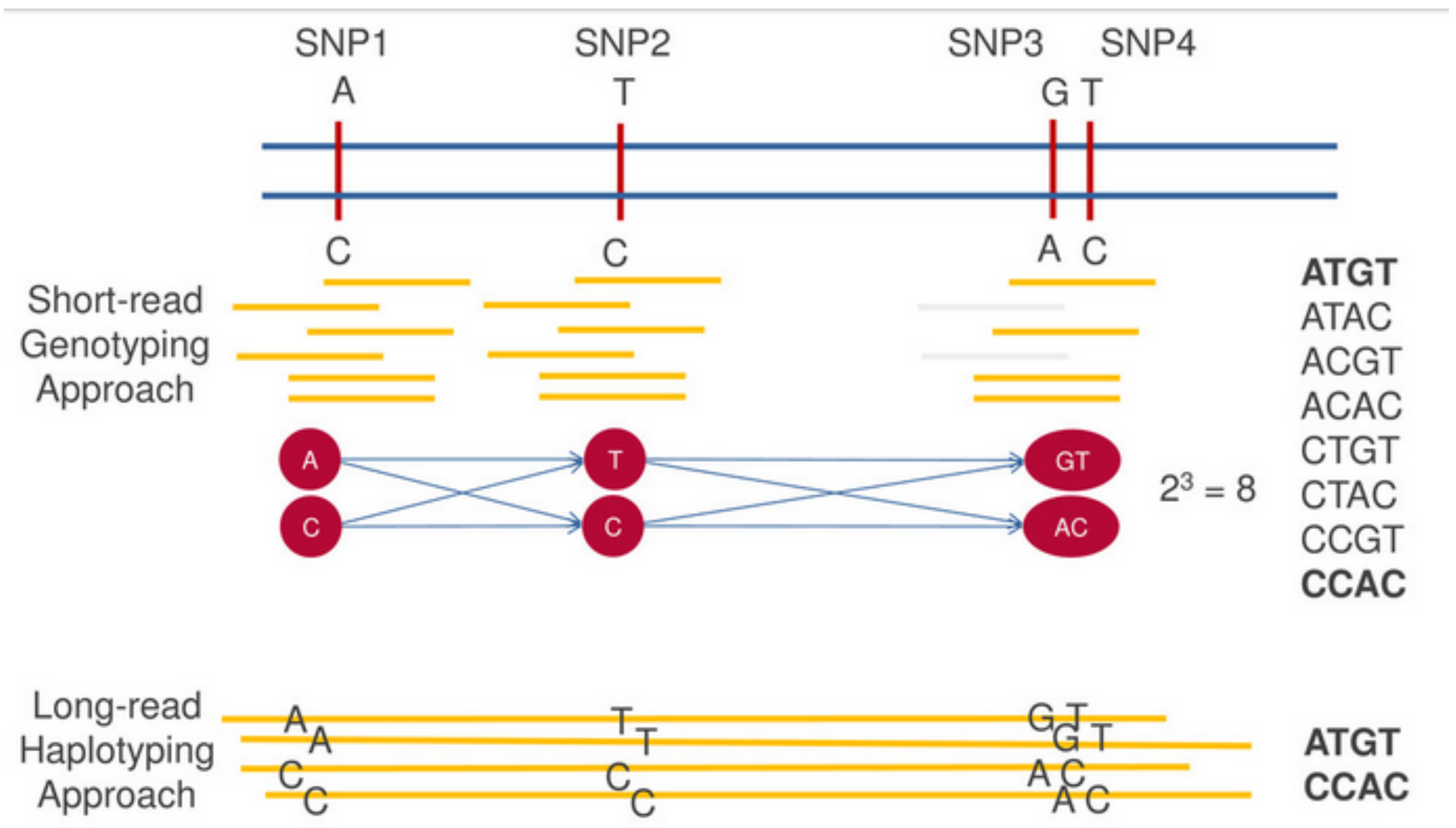
Allele 2: TCCTGTC

Or do we have a homozygous?

Allele 1: TCCTGTG

Allele 2: TCCTGTG

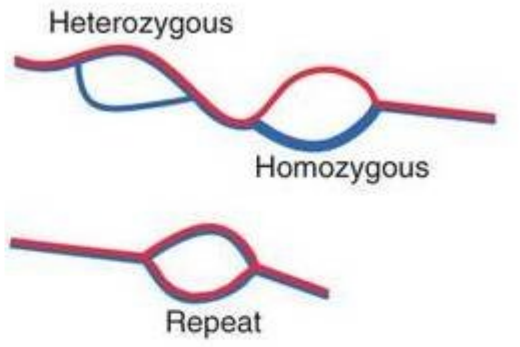
Haplotype issue calling – Long reads to the rescue



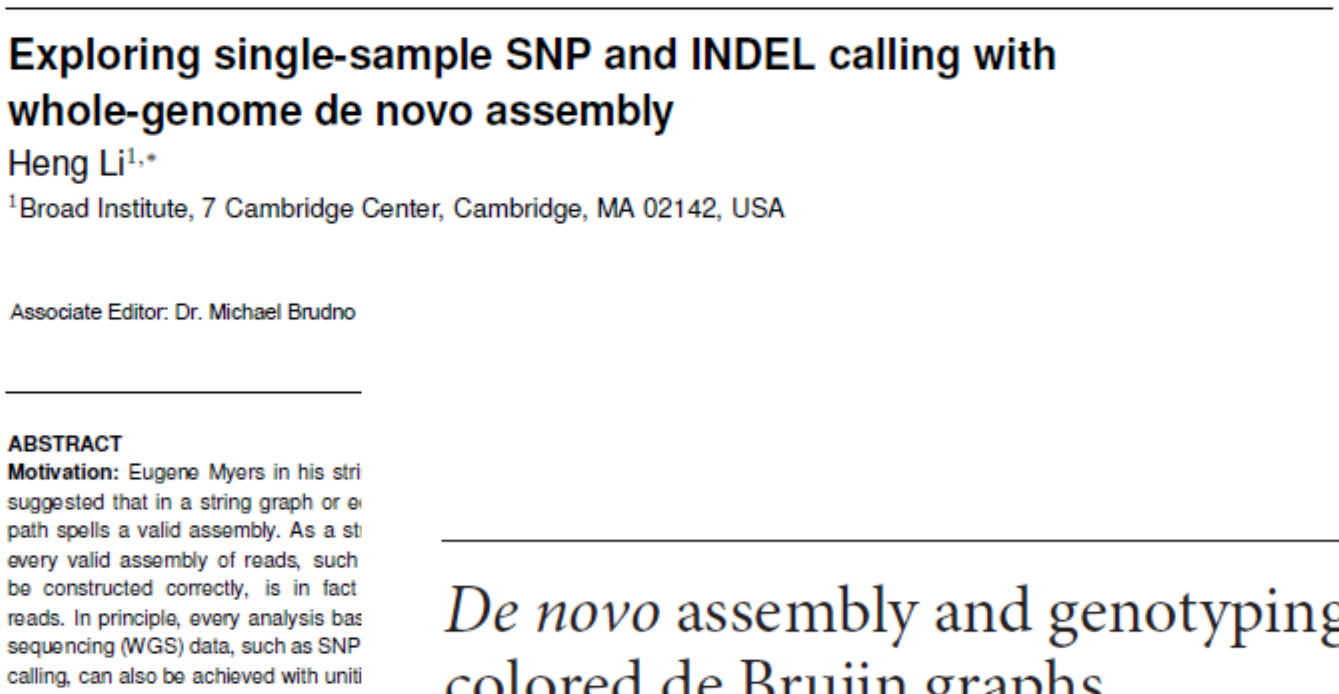
New methods of SNP calling

- Why align at all?
  - We only do this because of computational constraints
  - Ideally we want to assemble denovo and then align to reference genome

- Fermi and Cortex are tools to enable this:
  - Denovo genome assembler, but keeps track of differences which could be due to SNPs/Indels



Variant calling with de-novo assembly



Exploring single-sample SNP and INDEL calling with whole-genome de novo assembly

Heng Li[1,*]

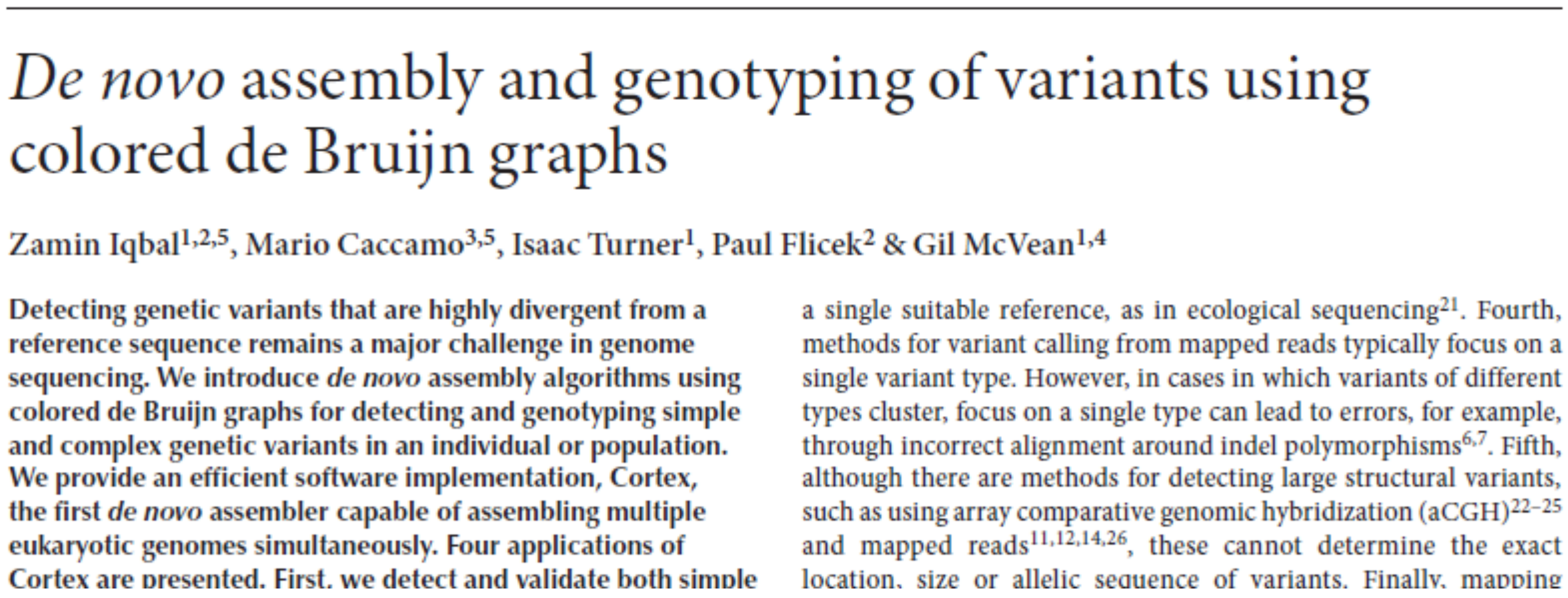
[1]Broad Institute, 7 Cambridge Center, Cambridge, MA 02142, USA

Associate Editor: Dr. Michael Brudno

ABSTRACT

Motivation: Eugene Myers in his stri
suggested that in a string graph or e
path spells a valid assembly. As a st
every valid assembly of reads, such
be constructed correctly, is in fact
reads. In principle, every analysis bas
sequencing (WGS) data, such as SNP
calling, can also be achieved with uniti

nature genetics



De novo assembly and genotyping of variants using colored de Bruijn graphs

Zamin Iqbal[1,2,5], Mario Caccamo[3,5], Isaac Turner[1], Paul Flicek[2] & Gil McVean[1,4]

Detecting genetic variants that are highly divergent from a reference sequence remains a major challenge in genome sequencing. We introduce *de novo* assembly algorithms using colored de Bruijn graphs for detecting and genotyping simple and complex genetic variants in an individual or population. We provide an efficient software implementation, Cortex, the first *de novo* assembler capable of assembling multiple eukaryotic genomes simultaneously. Four applications of Cortex are presented. First, we detect and validate both simple

a single suitable reference, as in ecological sequencing[21]. Fourth, methods for variant calling from mapped reads typically focus on a single variant type. However, in cases in which variants of different types cluster, focus on a single type can lead to errors, for example, through incorrect alignment around indel polymorphisms[6,7]. Fifth, although there are methods for detecting large structural variants, such as using array comparative genomic hybridization (aCGH)[22–25] and mapped reads[11,12,14,26], these cannot determine the exact location, size or allelic sequence of variants. Finally, mapping

Questions!