Burrows-Wheeler Transformation and Short-Read Alignment

BIOL 435/535: Bioinformatics
March 31, 2022

Needleman & Wunsch Global Pairwise Alignment Matrix construction

Rules:

Start at origin at 0

Match (diagonal) = +1

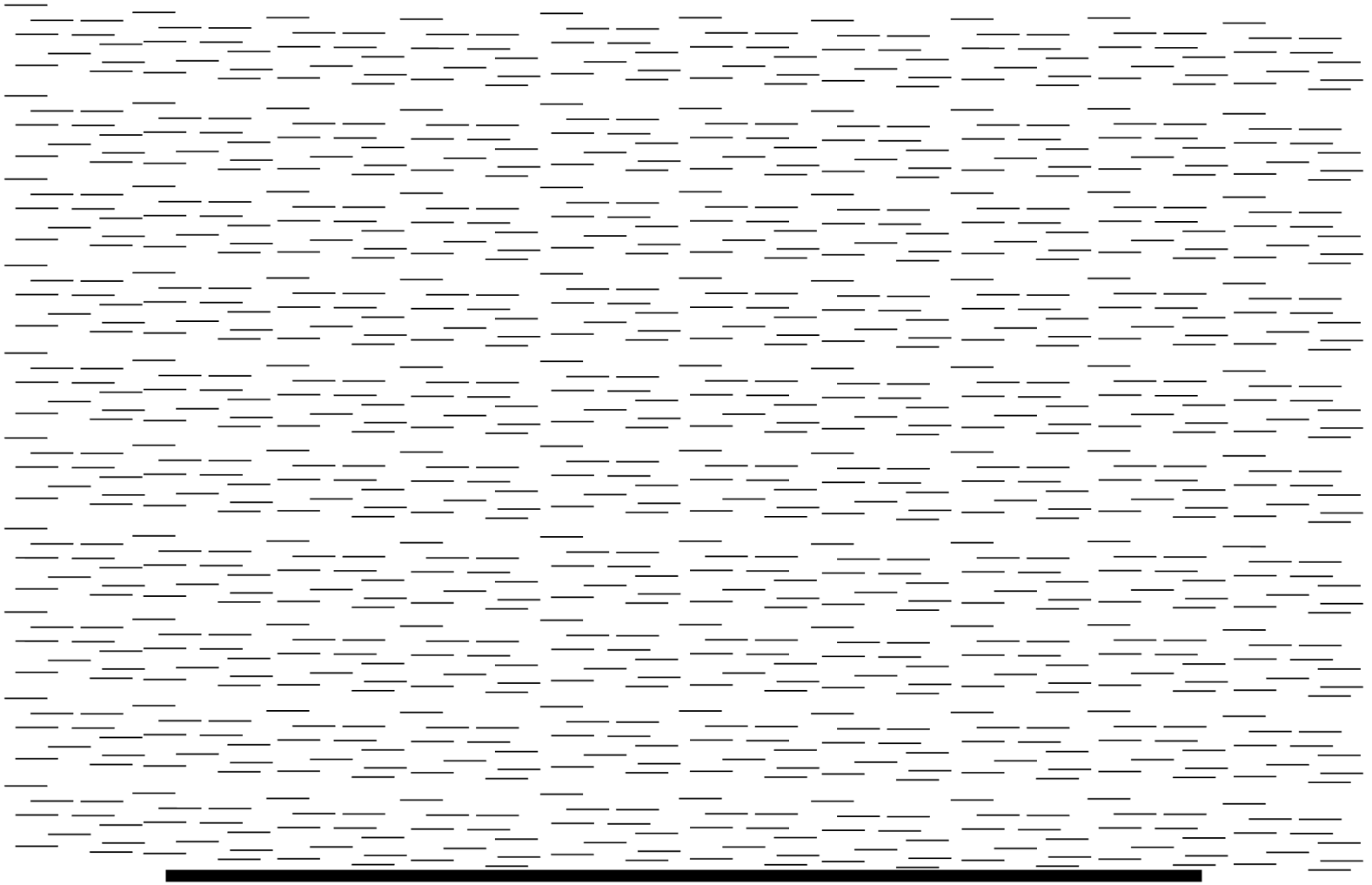
Mismatch (diagonal) = 0

Gap (right or down) = -1

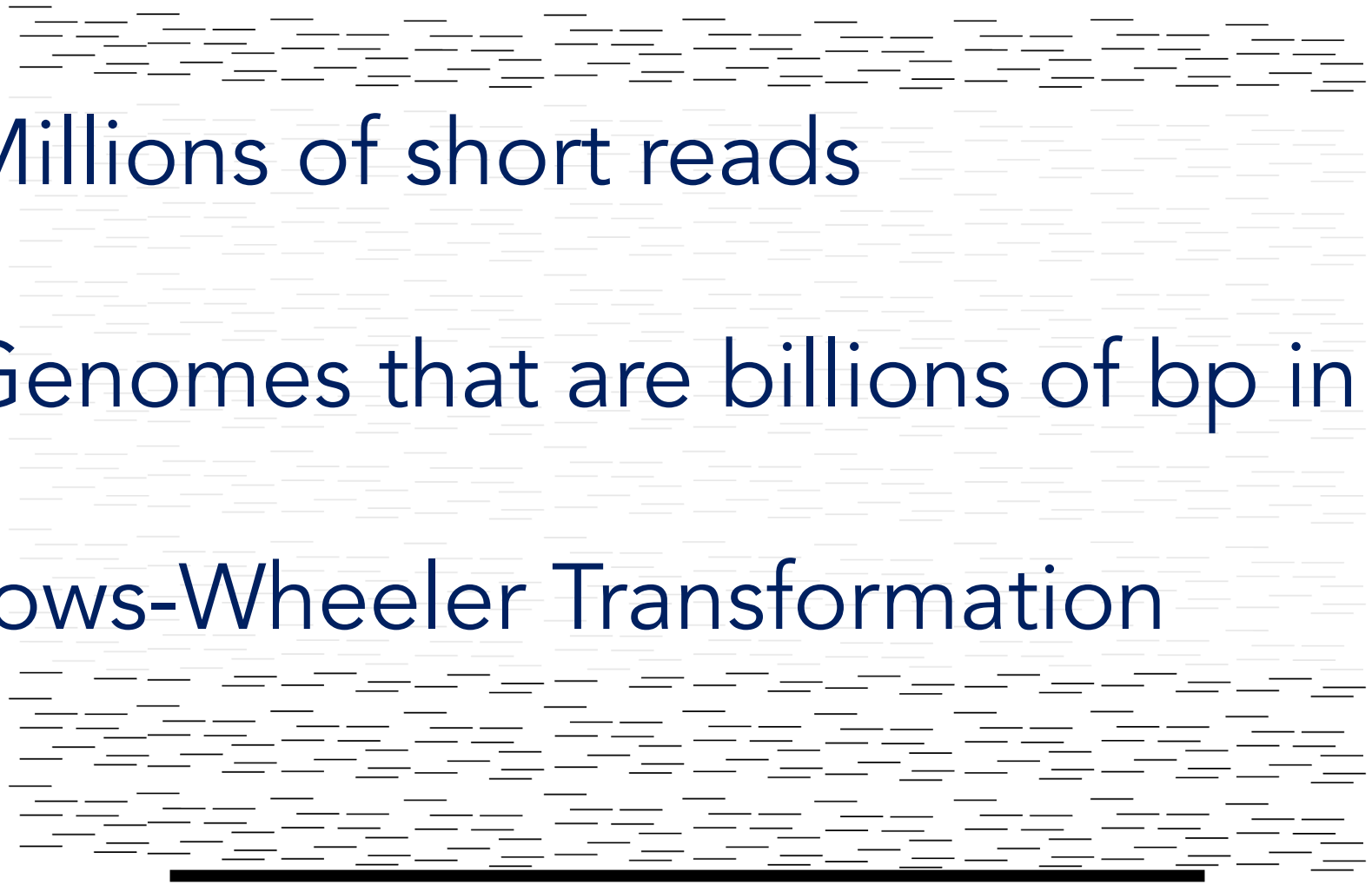
Each cell gets highest possible score from three adjacent cells

		A	B	C	N	J	R	O	C	L	C	R	P	M
	0													
A														
J														
C														
J														
N														
R														
C														
K														
C														
R														
B														
P														

Pairwise alignment is easy, but…



Pairwise alignment is easy, but…



1) Millions of short reads

2) Genomes that are billions of bp in length

Burrows-Wheeler Transformation

Burrows-Wheeler Transformation

- Method for compressing text strings
- Transforms text strings into a “suffix array”
- Reversible, fast (~linear), not too computationally intensive
- Reduces the search space by a factor of the number of characters

Burrows-Wheeler Transformation

Google$

Burrows-Wheeler Transformation

1) Generate all substrings

Google$

Burrows-Wheeler Transformation

1) Generate all substrings

Google$
$Google
e$Googl
le$Goog
gle$Goo
ogle$Go
oogle$G

Burrows-Wheeler Transformation

2) Sort the substrings alphabetically

Google$		e$Googl
$Google		gle$Goo
e$Googl		Google$
le$Goog	→	le$Goog
gle$Goo		ogle$Go
ogle$Go		oogle$G
oogle$G		$Google

Burrows-Wheeler Transformation

3) Last column (i.e., suffix) contains all the necessary information of all the subsequences

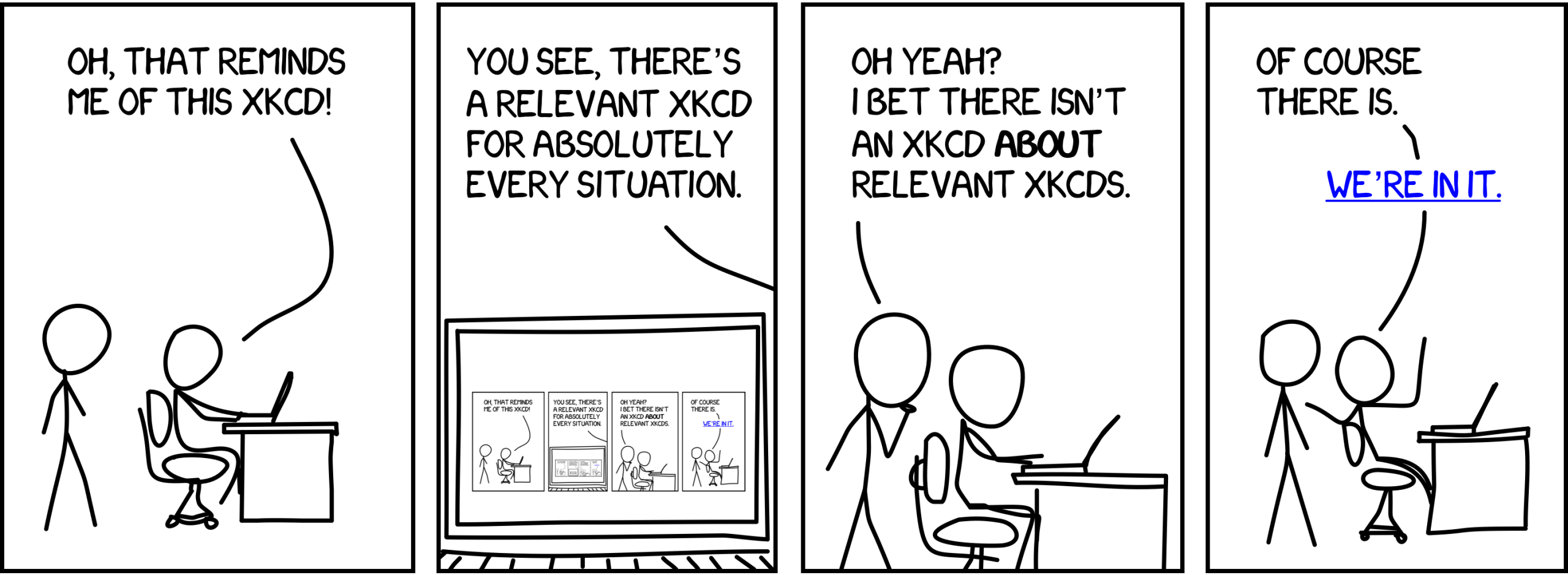
e$Googl
gle$Goo
Google$
le$Goog
ogle$Go
oogle$G
$Google

Burrows-Wheeler Transformation

4) Recursively store the suffixes as a hash table (array, dictionary, etc) for quick retrieval

lo$goGe

Recursion: Use a program to solving a problem by calling on the same program



Recursion: Use a program to solving a problem by calling on the same program

1. Base Case (i.e., when to stop)

2. Work toward Base Case

3. Recursive Call (i.e., call it again)

Burrows-Wheeler Transformation

4) Recursively store the suffixes as a hash table (array, dictionary, etc) for quick retrieval

Short-read aligners

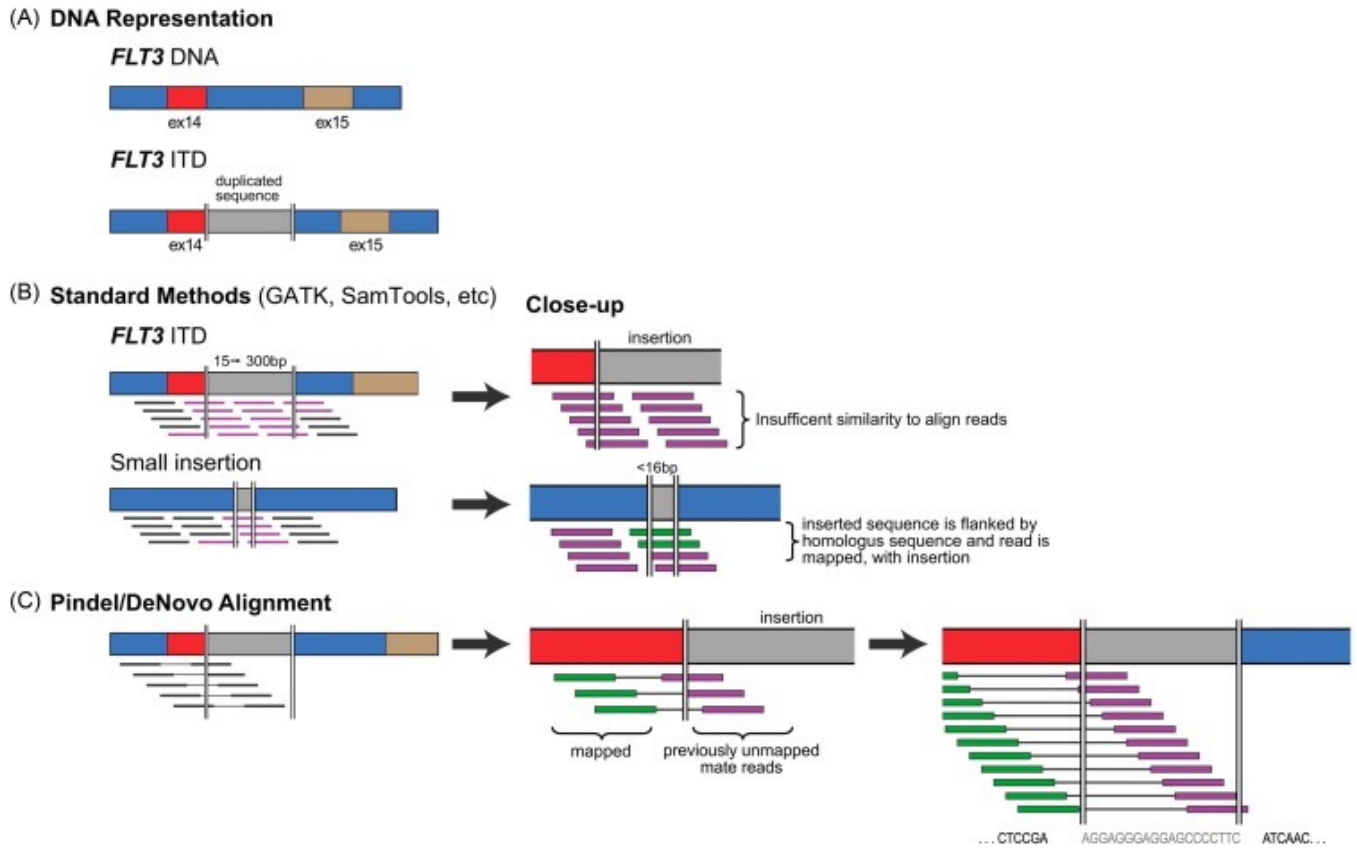
- BWA

```
bwa mem –apt 24 reference.fasta reads.fastq.gz > out.sam

bwa mem –apt 24 reference.fasta reads.fastq.gz |
     \samtools view -F 4 - |
     \samtools sort -@ 24 -o BAM - > out.mapped.sorted.bam
```

- Bowtie

Interpreting Read Alignments



SAM/BAM Format

- Sequence Alignment Map/ Binary Alignment Map
- Tab-delimited descriptions of reads aligned to a reference
- Header section
  - Describes the reference, the mapping tool, the alignment parameters, the reads that were aligned., sequencing chemistry, etc.
- Body section
  - Describes the individual alignments

Samfile body section

Col	Field	Type	Regexp/Range	Brief description
1	QNAME	String	`[!-?A-~]{1,254}`	Query template NAME
2	FLAG	Int	$[0,2^{16}-1]$	bitwise FLAG
3	RNAME	String	`\*|[!-()+-<>-~][!-~]*`	Reference sequence NAME
4	POS	Int	$[0,2^{31}-1]$	1-based leftmost mapping POSition
5	MAPQ	Int	$[0,2^{8}-1]$	MAPping Quality
6	CIGAR	String	`\*|([0-9]+[MIDNSHPX=])+`	CIGAR string
7	RNEXT	String	`\*|=|[!-()+-<>-~][!-~]*`	Ref. name of the mate/next read
8	PNEXT	Int	$[0,2^{31}-1]$	Position of the mate/next read
9	TLEN	Int	$[-2^{31}+1,2^{31}-1]$	observed Template LENgth
10	SEQ	String	`\*|[A-Za-z=.]+`	segment SEQuence
11	QUAL	String	`[!-~]+`	ASCII of Phred-scaled base QUALity+33

QNAME = Read name

FLAG interpretation

- Bitwise flag describing the alignment
  - 0 – Aligned forward strand
  - 16 – Aligned reverse strand
  - 4 – Read not aligned
  - 256 – Secondary alignment – read aligns equally well to multiple locations
  - 1024 – PCR/optical duplicate
  - 2048 – Supplementary alignment – read aligns to multiple locations, with clipping

- Add up all the applicable flags to get the full bitwise flag

Interpreting sam flags

samtools view -F 256 input.sam > primary.sam

RNAME

- Reference sequence name (which chromosome did the read map to?)

- If the read is unmapped, will be empty “.”

POS

- 1-based leftmost start position of the alignment (regardless of orientation)
  - 1-based = (1:N), where N = last position you're interested in
  - 0-based = (0:N+1), where N = last position you're interested in

MAPQ = Mapping quality > 10

- Will be 0 for all non-primary alignments

CIGAR String

Describes the alignment

Op	BAM	Description
M	0	alignment match (can be a sequence match or mismatch)
I	1	insertion to the reference
D	2	deletion from the reference
N	3	skipped region from the reference
S	4	soft clipping (clipped sequences present in SEQ)
H	5	hard clipping (clipped sequences NOT present in SEQ)
P	6	padding (silent deletion from padded reference)
=	7	sequence match
X	8	sequence mismatch

CIGAR String

151M = 151 matching base pairs

90M1D61M = 90 matching base pairs, a deletion in the read relative to the reference, followed by 61 matching base pairs

40M1025N85M = 40 matching base pairs, a 1,025 bp intron, followed by 85 matching base pairs

Pacbio CIGAR String:

5845S23=1X13=1D11=1I1X8=1D19=1D6=1D2=1D4=1D6=1D5=4I5=1I33=1D35=1I1=1X19=1D7=1D2=1X1=1X5=1X16=1I7=30I2=2I6=1X5=1X12=1D10=1X8=1X7=1X4=1X1=1X18=1X3=1D3=1X3=1I4=2I9=3I1=1I1=3I4=1I2=3I3=2I2=1I6=1X8=1X10=1X4=1D4=1X5=1D14=1D1=1X19=1D2=1X10=1I22=1I3=1D15=1D10=1I2=1D30=1I29=1X10=2I2=1I15=1D6=1D3=1D9=1I2=1I1=1D11=2I1=1I18=1D19=1I26=1D28=1D3=1X1=1D16=1D4=1I4=6I3=1I22=1D2=1D31=2D43=1I4=1D2=1X3=1I8=1I24=1D18=1X14=1X18=1X2=1I4=1I31=1I5=1X6=1I19=1D19=1I28=1X11=2D9=1X9=1X11=1X2=1D1=1X5=1D1X20=1I3=1D4=1D6=1D3=1I32=2I27=1D5=1D7=1X8=1D27=1X3=1D1=1I1=1I1X1=1I4=1I13=1I13=1I1=1X11=1X3=1I10=1D4=1X25=1D8=2I4=1D10=1D4=2D8=1I19=1D4=1X6=1D8=1X39=1D9=1D1=1X3=1I8=1I1X5=1I9=2I8=1D7=1X7=1I2=1I20=1I10=1I16=1I22=1D5=1X4=1D5=1D8=2I1=1X5=1D7=1I18=1X21=1I15=1D4=1I15=1I3=1D4=1X3=1X31=1I28=1D42=1I9=1D4=1X6=1X10=1D30=1D30=2I24=2I11=1X6=1X4=2D10=1X9=1X13=2I8=6I8=1D39=1I6=1D2=1D7=1I10=1D4=1D6=1X2=2D21=1D33=5I9=1I6=1X1=1I9=1D12=1D8=1D32=1I12=1D2=1D4=1I1=1D7=1I17=1D8=1I19=1D5=1D8=1X5=1I7=1I16=1D3=1D9=1I11=1I17=1I1X26=3I9=1D9=1I7=4I10=1I18=4I1=1X6=1I4=1D10=1I2=1I6=1I4=1D4=3I4=1I1=2I4=1D6=1D32=1I6=1D5=1X10=1D13=1I5=1I2=1I5=1D24=1I1X17=1I3=1I1X11=1X27=1D7=1X17=1I8=1D22=1I2=1I26=1I5=1D32=1D5=1I6=2I6=1I10=1D22=1I8=1X12=1I7=1D5=2D11=1I4=1D1=1X36=1X10=1D24=1X37=1D9=1D2=1D3=1D5=1D1=1D2=1D1=2D5=1X9=1D9=1D12=1X2=1X5=1D2=1I13=1D5=1D17=1D10=1D3=1D19=1X13=1I4=1I9=1D1=1I4=1X9=1I5=1X5=1X2=1X8=1D32=1X27=1D21=1X1=1D4=1D3=1X8=1D22=1D1=1I6=1I2=3I2=1D6=1I5=1I6=1D18=1I2=1I13=4I2=1I3=1I16=1I17=5I21=1X16=1D7=1I12=1I7=1I7=1D11=2I5=2I23=1D8=1D5=1I16=1D10=1D5=2X3=1I3=1D5=1X19=2I14=1I6=1I9=1D14=1D2=1D19=1D28=1I14=1D3=1D2=1D8=1I6=1D22=1I3=1D7=1D1=1I2=1I12=1I1=1I5=1D6=1D12=1X15=1D1X6=1X10=2I8=1I6=1X21=1I16=1D2=1X7=1I11=1I4=1I14=1I4=1D2=1I3=1D2=1I3=2X2=1I3=1I18=1D5=1D7=1I6=1D11=1D26=1D1X19=1D4=1I9=2I1X2=2I7=1I1=4I1=1I15=1I6=1X3=1X23=1D13=1I8=1I1=1I7=1X13=1I18=1X6=1I1=1X2=1I4=1D16=1D11=1I1X17=1D15=1D10=1X13=1D2=1D15=1D8=1D26=1D10=1X3=1I22=1X6=1X7=1I8=4I1=1I18=1X14=1D6=1I4=1I30=1X8=1D4=3I1=1I8=3I2=2I24=4I26=1D4=1I4=1I4=1X6=1I11=1I1=2I2=1X3=1X3=1I6=2I3=2I1=6I1X1=1I1=1I2=1I1=2I2=1I1=5I1X3=1I2=1X5=1D5=1X7=1D4=1X10=1D4=1D21=1I20=1I7=1X10=1D23=1I1X1=1I20=1X15=1I4=7I2=1I9=1I9=3I1=1X2=1I1=2I19=1D8=1D2=1I1=1D8=1D8=1D10=1D17=1D24=1I6=1D23=1D24=1D6=2X5=1D2=1I10=1I2=1D17=2D16=1D1X3=1D1=1X4=1I9=1D3=1D8=1D10=1X2=1D7=1D2=1I2=1D2=1X12=1D8=1X16=1D1X13=1I2=1I9=1I2=1X1=2I22=1X1=1D30=1X29=1D20=1I4=1I4=1I46=1I10=1X1=1I3=1D35=1X7=1I6=1D7=1I3=1D21=1D42=1I2=1I7=1I15=1I4=2I1=1I4=1X2=1X14=1X21=1D6=1D13=1I1X11=1I5=1D5=1X18=1I1X18=1X3=1D14=1X12=1D15=1D7=1D5=1X8=1D16=1X8=1I12=1D6=1I23=1D5=1I10=1I2=1I6=1D7=1I7=1X8=1D10=1X16=2I5=1X6=1D8=1I3=1I1=1I4=1I15=1D16=1X2=1I11=1I4=2I10=1X6=1I2=1X13=1X19=1X3=1D12=1D2=1I44=1I5=1I5=1I7=1D9=1X6=1I24=1X6=1X17=1I3=2X4=1D5=2I1=2I3=1D7=1X3=1D5=1D7=1D14=2I7=1D3=1X6=1D2=1D10=1X1=1I10=2I2=1X2=1X7=1D1X4=1X7=1X3=1D1=1X5=1D1=1X21=1D5=1D5=3X4=1X3=2D3=2X1=2X1=1X2=1X2=1X1=1X10=2I1=2X1=4X2=2X2=1X3=6X3=1I1=1X2=1X1=1X1=1X10=1X2=2X3=1I6=1I1=1X4=1X1=1X1=3X1=1X4=1I1X5=2X1=2X1=1X2=1I17=1D4=1D3=1X16=1X11=1I1X12=1D10=1D20=1I34=1X1=1D10=1D28=1I3=1I8=1X6=1X8=1I6=1I3=1D25=1D2=1D2=1D5=1X31=1I2=1X12=1D4=1I15=1I7=1X10=1D5=1X7=1D1=1D10=1D9=1I3=2I10=1I12=1D6=1X42=2I41=1D3=1D17=1D11=1D4=1X2=1X8=1D9=1I11=1I1=1I1=2I8=1I7=1X17=1I1=1D1=1X5=1I5=1I3=1I17=1I17=1D23=1D11=1I24=1X5=1I4=1X4=1I9=1I7=1I9=1I9=1I20=1D24=1D14=1D1X2=1D18=1I1=1X5=1D15=2D3=1D3=1D1X1=2D30=1I17=1D16=1X8=1D6=1X15=1X5=1I6=1X5=1D2=1D18=1I2=1D8=2D23=1I15=1X13=1D18=2I11=1I2=1X6=2I12=1I1=1D11=1I9=1X13=1X16=4I14=1X20=1D26=1X4=1D9=2X13=1I3=1D4=1X6=1I13=

CIGAR String

Clipping

- Soft vs. hard clipping
- Can be used to detect structural variants

RNEXT = reference sequence of the paired read/ mated read pair

PNEXT = Leftmost position of the paired read/ mated read pair

TLEN = Length of the reference sequence in the aligned region

SEQ – aligned sequence

QUAL – Per-base quality scores of the aligned nucleotides

Samtools

view – read sam/bam files, convert to BAM/CRAM

sort – sort sam files by alignment position or by read name

depth – get per-position depth information

mpileup – Pile up reads on individual nucleotides

merge – merge separate bam files into one

flagstat – Get basic statistics from your sam/bam file (needs to be indexed)

index – Create an index of the sam/bam file

faidx – Create an index of a fasta, extract sequence from fasta

Pileup

- Transform SAM/BAM file into VCF/BCF file format
- Piles up reads on individual nucleotides