Read Mapping and Variant Calling

Bioinformatics Workshop for *M. tuberculosis* Genomics and Phylogenomics

July 10-14, 2018 @The Philippine Genome Center

Ulas Karaoz, PhD

Ecology Department
Berkeley Lab

BERKELEY LAB
Lawrence Berkeley National Laboratory
Bringing Science Solutions to the World

https://eesa.lbl.gov/profiles/ulas-karaoz, Email: ukaraoz@lbl.gov, Twitter: @ukaraoz

Learning Objectives

- How read mappers work
- File formats involved in read mapping
- Querying and filtering read alignments
- Calling variants

What is Read Mapping?

genome

```
TGGCTTGCGGCCATCAACGGTTTCCTACCGAGGGGGCGGTGCGGGCATC
ACTAGTGCGGAAACTGGGAGGGCTCTCGGCCCTCCGCCTTTAGGCGGTG
CTTACCCCTTCGTAGAGGGGCTGTGCGGCCATCCCGCGAGGATCCGAGA
AGGCGAGCGTGCGGATCCCACCCGGGGGGACGGGCCTCAAAGCCGCCTT
```

read

```
TGCGGCCATC
```

What is Read Mapping?

genome

```
TGGCTTGCGGCCATCAACGGTTTCCTACCGAGGGGGCGGTGCGGGCATC
ACTAGTGCGGAAACTGGGAGGGCTCTCGGCCCTCCGCCTTTAGGCGGTG
CTTACCCCTTCGTAGAGGGGCTGTGCGGCCATCCCGCGAGGATCCGAGA
AGGCGAGCGTGCGGATCCCACCCGGGGGGACGGGCCTCAAAGCCGCCTT
```

read

```
TGCGGCCATC
```

read mapping

Find all genomic locations "`TGCGGCCATC`" might have been generated from

TGGCT**TGCGGCCATC**AACGGTTTCCTACCGAGGGGGCGG**TGCGGGCATC**
ACTAGTGCGGAAACTGGGAGGGCTCTCGGCCCTCCGCCTTTAGGCGGTG
CTTACCCCTTCGTAGAGGGGCTG**TGCGGCCATC**CCGCGAGGATCCGAGA
AGGCGAGCG**TGCGGATC**CCACCCGGGGGGACGGGCCTCAAAGCCGCCTT

What is Read Mapping?

genome

```
TGGCTTGCGGCCATCAACGGTTTCCTACCGAGGGGGCGGTGCGGGCATC
ACTAGTGCGGAAACTGGGAGGGCTCTCGGCCCTCCGCCTTTAGGCGGTG
CTTACCCCTTCGTAGAGGGGCTGTGCGGCCATCCCGCGAGGATCCGAGA
AGGCGAGCGTGCGGATCCCACCCGGGGGGACGGGCCTCAAAGCCGCCTT
```

read TGCGGCCATC

read mapping Find all genomic locations "TGCGGCCATC" might have been generated from

```
TGGCTTGCGGCCATCAACGGTTTCCTACCGAGGGGGCGGTGCGGGCATC
ACTAGTGCGGAAACTGGGAGGGCTCTCGGCCCTCCGCCTTTAGGCGGTG
CTTACCCCTTCGTAGAGGGGCTGTGCGGCCATCCCGCGAGGATCCGAGA
AGGCGAGCGTGCGGATCCCACCCGGGGGGACGGGCCTCAAAGCCGCCTT
```

- find all "approximate" occurrences
- analogous to string matching

Read Mapping isn't Read Alignment

read alignment

```
TGGCTTGCGGCCATCAACGGTTTCCTACCGAGGGGGCGGTGCGGGCATC
     ||||||||||                          ||||| ||||
     TGCGGCCATC                          TGCGGCCATC
ACTAGTGCGGAAACTGGGAGGGCTCTCGGCCCTCCGCCTTTAGGCGGTG

CTTACCCCTTCGTAGAGGGGCTGTGCGGCCATCCCGCGAGGATCCGAGA
                        ||||||||||
                        TGCGGCCATC
AGGCGAGCGTGCGG--ATCCCACCCGGGGGGACGGGCCTCAAAGCCGCCTT
     |||||   |||
     TGCGGCCATC
```

Read Mapping isn't Read Alignment

read alignment

```
TGGCTTGCGGCCATCAACGGTTTCCTACCGAGGGGGCGGTGCGGGCATC
     ||||||||||                          ||||| ||||
     TGCGGCCATC                          TGCGGCCATC
ACTAGTGCGGAAACTGGGAGGGCTCTCGGCCCTCCGCCTTTAGGCGGTG

CTTACCCCTTCGTAGAGGGGCTGTGCGGCCATCCCGCGAGGATCCGAGA
                        ||||||||||
                        TGCGGCCATC
AGGCGAGCGTGCGG--ATCCCACCCGGGGGGACGGGCCTCAAAGCCGCCTT
      |||||   |||
      TGCGGCCATC
```

sequence alignment ≠ read mapping ≠ read alignment

Placing Read Mappers into the bigger picture

- **Sensitive global aligners:**
  - Needleman-Wuncsh
  - Smith-Waterman
  - Pair-HMM
- **Pairwise heuristic:**
  - fasta
  - Blast
  - Blat
  - Exonerate

- **Whole genome:**
  - Mavid
  - Mummer
  - Mauve
  - Lagan
  - BlastZ
- **Short read:**
  - Maq
  - SHRIMP
  - ELAND
  - bowtie
  - bwa
  - bbmap
  - SOAP

Read Mapping: Two Algorithmic Approaches

Read Mapping

- **I. Hash indexing**
- **II. Burrows-Wheeler transform**

Read Mapping Approaches: I. Hash Indexing

Hash all genome *k*-mers using a hash table

key: all *k*-mers of fixed length *k*
value: genomic positions

Mapping Algorithms: Hashing

```
ACGGTATTGTACCACATCC
```

	hash positions
ACGGTATTGTA	1200
CGGTATTGTAC	1201,12340
GGTATTGTACC	1202,995,23400
GTATTGTACCA	1203,8010
TATTGTACCAC	1204
ATTGTACCACA	1205
TTGTACCACAT	1206
TGTACCACATC	1207,34012

hashes → → ↓

aligned position: 1200

Read Mapping Approaches: II. Burrows-Wheeler Transform

ACAACG$

all rotations

ACAACG$	$ACAACG
$ACAACG	AACG$AC
G$ACAAC	ACAACG$
CG$ACAA	ACG$ACA
ACG$ACA	CAACG$A
AACG$AC	CG$ACAA
CAACG$A	G$ACAAC

lexical sorting

$ < A < C < G < T

GC$AAAC

Observe: rotations are *exposing* the suffixes

all suffixes

ACAACG$	
$	$ACAACG
G$	AACG$AC
CG$	ACAACG$
ACG$	ACG$ACA
AACG$	CAACG$A
CAACG$	CG$ACAA
ACAACG$	G$ACAAC

How all this is useful???

Burrows M, Wheeler DJ, A block sorting lossless data compression algorithm. Digital Equipment.

Read Mapping Approaches: II. Burrows-Wheeler Transform

$ A C A A C **G**	$ A C A A C $\mathbf{G}_1$
A A C G $ A **C**	$\mathbf{A}_1$A C G $ A $\mathbf{C}_1$
A C A A C G **$**	$\mathbf{A}_2$C A A C G $\mathbf{\$}_1$
A C G $ A C **A**	$\mathbf{A}_3$C G $ A C $\mathbf{A}_1$
C A A C G $ **A**	$\mathbf{C}_1$A A C G $ $\mathbf{A}_2$
C G $ A C A **A**	$\mathbf{C}_2$G $ A C A $\mathbf{A}_3$
G $ A C A A **C**	$\mathbf{G}_1$$ A C A A $\mathbf{C}_2$

Can we reconstruct the original string from BWT?

```
LF(6, "C") = occ("C")+count(6,"C")

LF(6, "C") = occ("G")+count(6,"G")
```

Mapping Algorithms: BWT/Suffix Array

BWT: Burrows-Wheeler Transform

```
ACGGTATTGTACCACATCC
```

		candidate hits in BW index
	C	++++++++
look up suffixes →	TACCACATCC	+++
	ACGGTATTGTACCACATCC	+

↓

convert BW coordinates back to genomic location → **aligned position: 1200**

Mapping Quality

mapping quality: confidence in a read's genomic origin

- Probability of misalignment depends on:
  - uniqueness of the aligned region in the genome
  - alignment length
  - number of mismatches and gaps
- Expressed in Phred scores, similar to base qualities:
  - $AQ = -10 \log_{10}(P_{misaligned})$
- Values not standardized across different aligners. For instance, the value for "unique" mapping is not uniform across aligners.

Output Formats: SAM and BAM

SAM: Sequence Alignment/Map

Visualisation

Method: BAM viewer Redraw Restore

```
@HD     VN:1.4    SO:coordinate
@SQ     SN:chr1   LN:249250621
@SQ     SN:chr10  LN:135534747
@SQ     SN:chr11  LN:135006516
@SQ     SN:chr12  LN:133851895
@SQ     SN:chr13  LN:115169878
@SQ     SN:chr14  LN:107349540
@SQ     SN:chr15  LN:102531392
@SQ     SN:chr16  LN:90354753
@SQ     SN:chr17  LN:81195210
@SQ     SN:chr18  LN:78077248
@SQ     SN:chr19  LN:59128983
@SQ     SN:chr2   LN:243199373
@SQ     SN:chr20  LN:63025520
@SQ     SN:chr21  LN:48129895
@SQ     SN:chr22  LN:51304566
@SQ     SN:chr3   LN:198022430
@SQ     SN:chr4   LN:191154276
@SQ     SN:chr5   LN:180915260
@SQ     SN:chr6   LN:171115067
@SQ     SN:chr7   LN:159138663
@SQ     SN:chr8   LN:146364022
@SQ     SN:chr9   LN:141213431
@SQ     SN:chrM   LN:16571
@SQ     SN:chrX   LN:155270560
@SQ     SN:chrY   LN:59373566
@PG     ID:TopHat VN:2.0.9  CL:/opt/chipster/tools/tophat2/tophat -p 2 --read-mismatches 2 -a 8 -m 0 -i 70 -I 500000 -g 20 --library-type fr-unstranded
--transcriptome-index=/opt/chipster/tools/bowtie2/indexes/hg19.ti --no-novel-juncs /opt/chipster/tools/bowtie2/indexes/hg19 reads1.fq
HWI-EAS229_1:4:82:1371:1147   272   chr1   18378   1   2M6358N73M*   0   0
TCCTGCTGAAGATGTCTCCAGAGACCTTCTGCAGGTACTGAAGGGCATCCGCCATCTGCTGGACGGCCTCCTCTC   5661525416816488666(6(6(6261?8==(B=513);(/BB=141=>6>?=<=?B>9B?>BA<66>BA>BB3
CC:Z:chr15 MD:Z:40C34 XG:i:0   NH:i:3   HI:i:0   NM:i:1   XM:i:1   XN:i:0   XO:i:0   CP:i:102506354   AS:i:0   XS:A:-   YT:Z:UU
```

optional header section

alignment section: one line per read,
containing 11 mandatory fields, followed by optional tags

BAM: binary version of SAM

Output Formats: SAM and BAM

- The specification
  - http://samtools.sourceforge.net/SAM1.pdf

- The SAM format consists of two sections:
  - Header section: Used to describe source of data, reference sequence, method of alignment, etc.
  - Alignment section: Used to describe the read, quality of the read, and nature alignment of the read to a region of the genome

- BAM is a compressed version of SAM
  - Compressed using lossless BGZF format
  - Other BAM compression strategies are a subject of research. See 'CRAM' format for example

- BAM files are usually 'indexed'
  - A '.bai' file will be found beside the '.bam' file
  - Indexing aims to achieve fast retrieval of alignments overlapping a specified region without going through the whole alignments. BAM must be sorted by the reference ID and then the leftmost coordinate before indexing

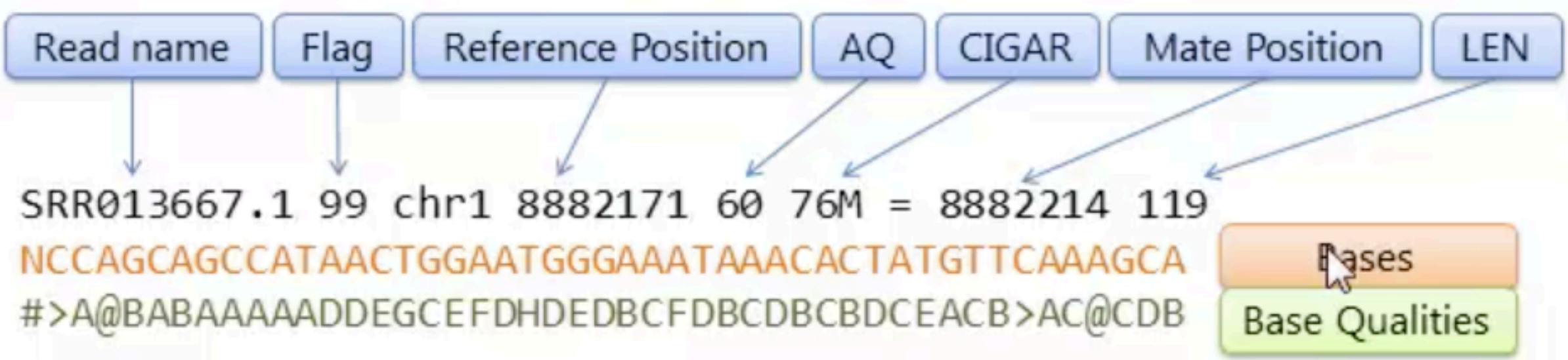
SAM/BAM header section

- Used to describe source of data, reference sequence, method of alignment, etc.
- Each section begins with character '@' followed by a two-letter record type code. These are followed by two-letter tags and values
  - @HD The header line
    - VN: format version
    - SO: Sorting order of alignments
  - @SQ Reference sequence dictionary
    - SN: reference sequence name
    - LN: reference sequence length
    - SP: species
  - @RG Read group
    - ID: read group identifier
    - CN: name of sequencing center
    - SM: sample name
  - @PG Program
    - PN: program name
    - VN: program version

https://samtools.github.io/hts-specs/SAMv1.pdf

SAM/BAM alignment section

Col	Field	Type	Regexp/Range	Brief description
1	QNAME	String	`[!-?A-~]{1,255}`	Query template NAME
★ 2	FLAG	Int	$[0,2^{16}-1]$	bitwise FLAG
3	RNAME	String	`\*|[!-()+-<>-~][!-~]*`	Reference sequence NAME
4	POS	Int	$[0,2^{29}-1]$	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 segment
8	PNEXT	Int	$[0,2^{29}-1]$	Position of the mate/next segment
9	TLEN	Int	$[-2^{29}+1,2^{29}-1]$	observed Template LENgth
10	SEQ	String	`\*|[A-Za-z=.]+`	segment SEQuence
11	QUAL	String	`[!-~]+`	ASCII of Phred-scaled base QUALity+33



https://samtools.github.io/hts-specs/SAMv1.pdf

SAM/BAM flag

- http://broadinstitute.github.io/picard/explain-flags.html
- 12 bitwise flags describing the alignment
- These flags are stored as a binary string of length 11 instead of 11 columns of data
- Value of '1' indicates the flag is set. e.g. 00100000000
- All combinations can be represented as a number from 1 to 2048 (i.e. 2^{11}-1). This number is used in the BAM/SAM file. You can specify 'required' or 'filter' flags in samtools view using the '-f' and '-F' options respectively

Bit		Description
1	0x1	template having multiple segments in sequencing
2	0x2	each segment properly aligned according to the aligner
4	0x4	segment unmapped
8	0x8	next segment in the template unmapped
16	0x10	SEQ being reverse complemented
32	0x20	SEQ of the next segment in the template being reverse complemented
64	0x40	the first segment in the template
128	0x80	the last segment in the template
256	0x100	secondary alignment
512	0x200	not passing filters, such as platform/vendor quality controls
1024	0x400	PCR or optical duplicate
2048	0x800	supplementary alignment

Note that to maximize confusion, each bit is described in the SAM specification using its hexadecimal representation (i.e., '0x10' = 16 and '0x40' = 64).

https://samtools.github.io/hts-specs/SAMv1.pdf

CIGAR String

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

- The CIGAR string is a sequence of base lengths and associated 'operations' that are used to indicate which bases align to the reference (either a match or mismatch), are deleted, are inserted, represent introns, etc.
- e.g. 81M859N19M

https://samtools.github.io/hts-specs/SAMv1.pdf

BED format: specify genomic regions as coordinates

- When working with BAM files, it is very common to want to examine a focused subset of the reference genome
- These subsets are commonly specified in 'BED' files
  - https://genome.ucsc.edu/FAQ/FAQformat.html#format1
- Many BAM manipulation tools accept regions of interest in BED format
- Basic BED format (tab separated):
  - Chromosome name, start position, end position
  - Coordinates in BED format are 0 based

Manipulation of BAM, SAM and BED files

- Several tools are used ubiquitously to query and manipulate these files
- SAM/BAM files
  - samtools
  - bamtools
  - picard
- BED files
  - bedtools
  - bedops

Sorting SAM/BAM

- Generally, BAM files are sorted by position
  - For computational performance reasons, when sorted and indexes, arbitrary access is much faster
- Certain tools require a BAM sorted by *"read name"*
  - Usually, when you need to easily identify both reads of a pair
  - Example: check is the insert size looks right

Alignment QC

Alignment Level QC

- How many reads mapped to the reference?
  - How many mapped uniquely?
- How many pairs mapped?
  - How many pairs mapped concordantly?
- Mapping quality distribution
- Quick and dirty (*samtools flagstat*)

```
52841623 + 0 in total (QC-passed reads + QC-failed reads)
0 + 0 duplicates
52841623 + 0 mapped (100.00%:-nan%)
52841623 + 0 paired in sequencing
28919461 + 0 read1
23922162 + 0 read2
42664064 + 0 properly paired (80.74%:-nan%)
44904884 + 0 with itself and mate mapped
7936739 + 0 singletons (15.02%:-nan%)
999152 + 0 with mate mapped to a different chr
357082 + 0 with mate mapped to a different chr (mapQ>=5)
```

- Detailed: *qualimap*

Duplicate reads (fragments)

Duplicate reads (fragments)

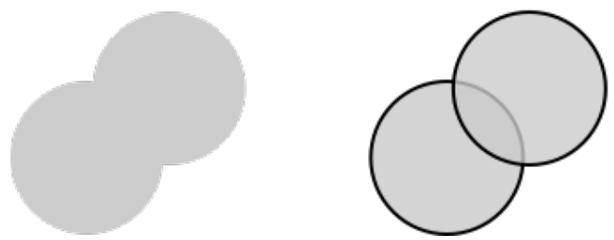
- If detected, essential to remove them before variant calling

- **Optical duplicates** (hint: think how Illumina sequencing by synthesis works)
  - generated when a single cluster of reads is part of two adjacent tiles' on the same slide and used to compute two read calls separately
  - very similar in sequence (except sequencing errors)
  - identified where the first, say, 50 bases are identical between two reads and the read's coordinates are close

- **Library duplicates** (only relevant for targeted sequencing where there is pre-library PCR)
  - generated when the original sample is preamplified to such extent that initial unique targets are PCR replicated prior to library preparation and will lead to several independent spots on the Illumina slide.
  - do not have to be adjacent on the slide
  - share a very high level of sequence identity
  - align to the same place on reference
  - identified from alignment to reference

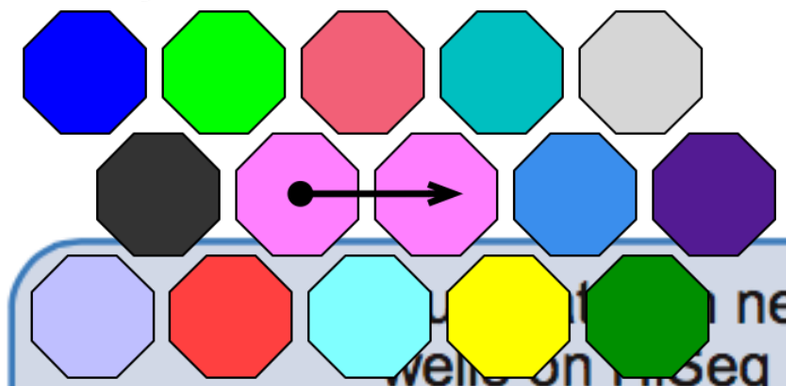
Duplicate Reads (Illumina)

"Optical" duplicates: caused by Illumina's optical detection/image processing



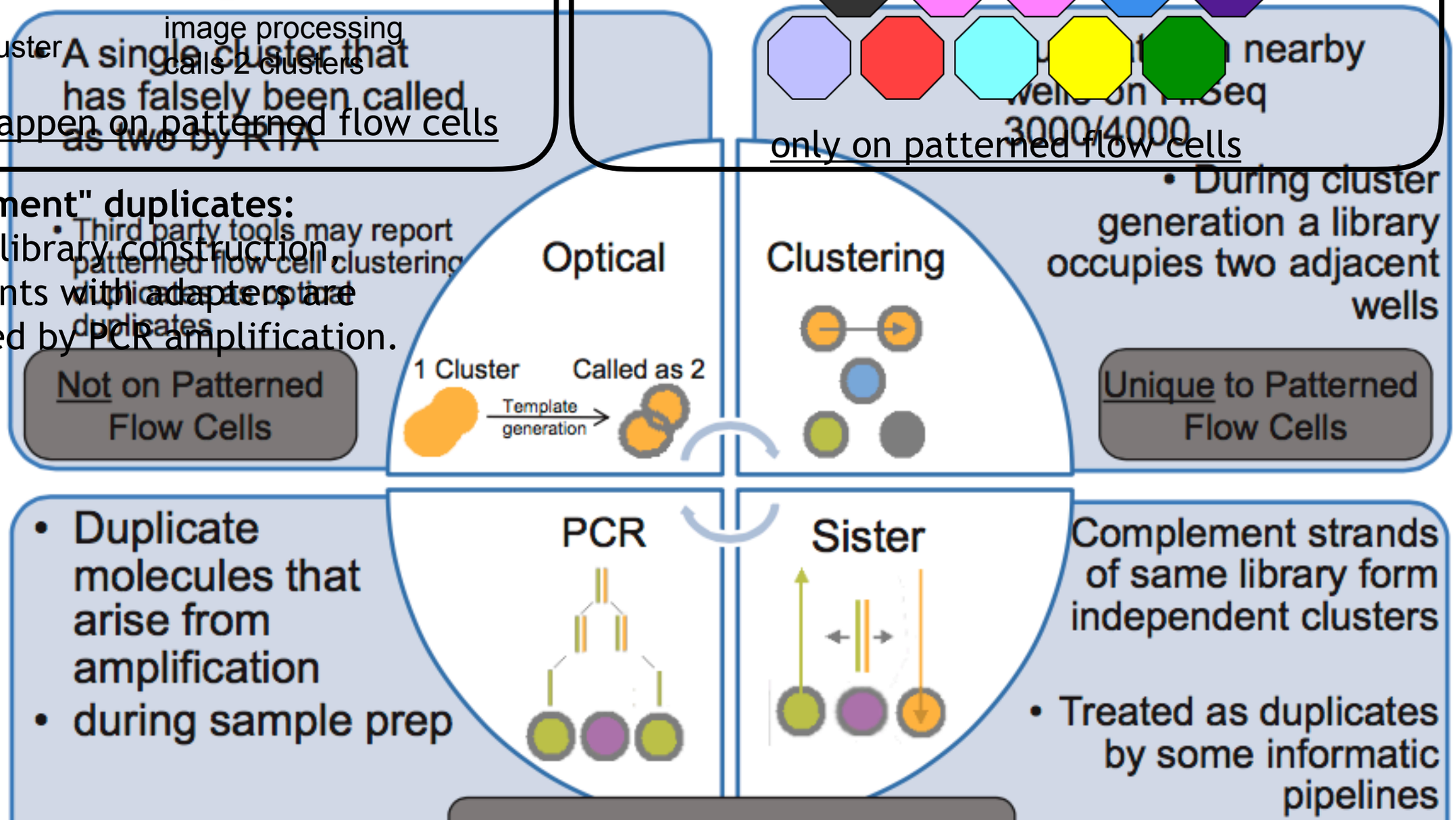
doesn't happen on patterned flow cells

"Exclusion Amplification (ExAmp)" duplicates: During **cluster generation,** a single cluster might occupy two adjacent wells.



only on patterned flow cells

"PCR/enrichment" duplicates: During library construction, fragments with adapters are enriched by PCR amplification.



Mappers are less accurate around indel sites

Solution: do local realignment post mapping

```
Reference  CTTTAGTTTCTTTT----CTTTCTTTCTTTCTTTTTTTTTAAGTCTCCCTC

           CTTTAGTTTCTTTT----GCCGCTTTCTTTCTTTCTT
           CTTTAGTTTCTTTT----GCCGCTTTCTTTCTTTCTT
Reads      CTTTAGTTTCTTTTGCCGCTTTCTTTCTTTCTTTTTTTTTAAGTCTCCCTC
           CTTTAGTTTCTTTTGCCGCTTTCTTTCTTTCTTTTTTTTTAAGTCTCCCTC
           CTTTAGTTTCTTTTGCCGCTTTCTTTCTTTCTTTTTTTTTAAGTCTCCCTC
           CTTTAGTTTCTTTTGCCGCTTTCTTTCTTTCTTTTTTTTTAAGTCTCCCTC
```

```
Reference  CTTTAGTTTCTTTT----CTTTCTTTCTTTCTTTTTTTTTAAGTCTCCCTC

           CTTTAGTTTCTTTTGCCGCTTTCTTTCTTTCTT
           CTTTAGTTTCTTTTGCCGCTTTCTTTCTTTCTT
Reads      CTTTAGTTTCTTTTGCCGCTTTCTTTCTTTCTTTTTTTTTAAGTCTCCCTC
           CTTTAGTTTCTTTTGCCGCTTTCTTTCTTTCTTTTTTTTTAAGTCTCCCTC
           CTTTAGTTTCTTTTGCCGCTTTCTTTCTTTCTTTTTTTTTAAGTCTCCCTC
           CTTTAGTTTCTTTTGCCGCTTTCTTTCTTTCTTTTTTTTTAAGTCTCCCTC
```

Ambiguous alignment around adjacent SNPs

Reference

AAGCGTCG

Reads

AAGCGTCG
AAGCGTCG
AAGCTACG
AAGCTACG
AAGCTACG

3 adjacent SNPs OR an insertion

Reference

AAG---CGTCG

Reads

AAG---CGTCG
AAG---CGTCG
AAG---CGTCG
AAGCTACG
AAGCTACG
AAGCTACG

Ambiguous alignment homopolymer runs flanked by adjacent SNPs

Reference

```
...CCCATTTTTTTCTAAAAGCTGGCAT...
```

Reads

```
   CCCATTTTTTTCTAAAAGCTGGCAT...
   CCCATTTTTTTCTAAAAGCTGGCAT...
...CCCATTTTTTTCTAAAA
...CCCATTTTTTTCTAAAA
...CCCATTTTTTTCTAAAA
```

Reference

```
...CCCATTTTTTTCTAAAAGCTGGCAT...
```

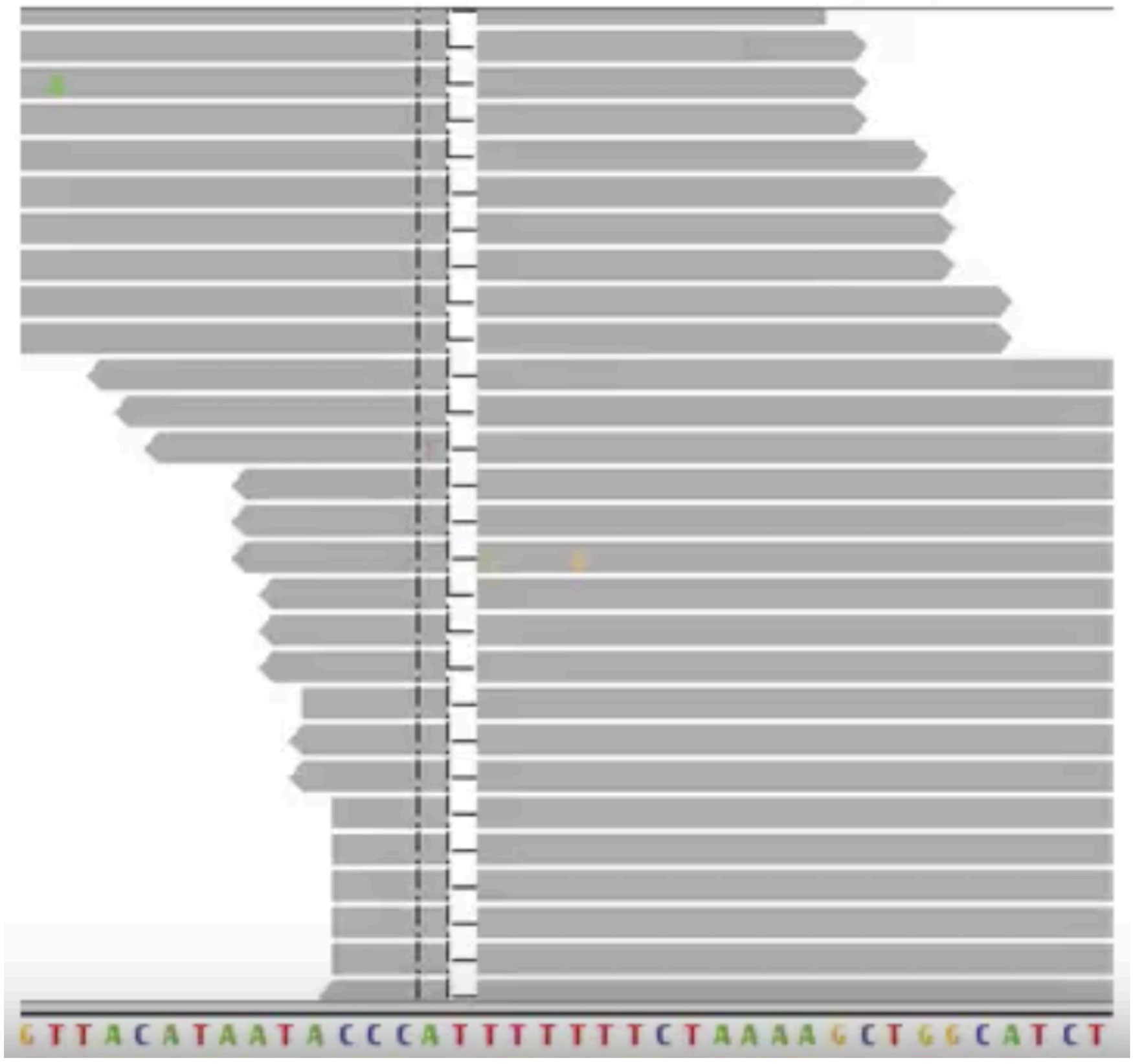
Reads

```
   CCCA-TTTTTTTCTAAAAGCTGGCAT...
   CCCA-TTTTTTTCTAAAAGCTGGCAT...
...CCCA-TTTTTTTCTAAAA
...CCCA-TTTTTTTCTAAAA
...CCCA-TTTTTTTCTAAAA
```

Alignment post-processing: Indel realignment

Alignment post-processing: Indel realignment

Alignment post-processing: Indel realignment



What makes a good aligner?

- Speed
- Accuracy
  - Novoalign
  - Razers3
- All-round
  - bwa/bwa-mem
  - bowtie
- Functionality
  - STAR
  - TopHat

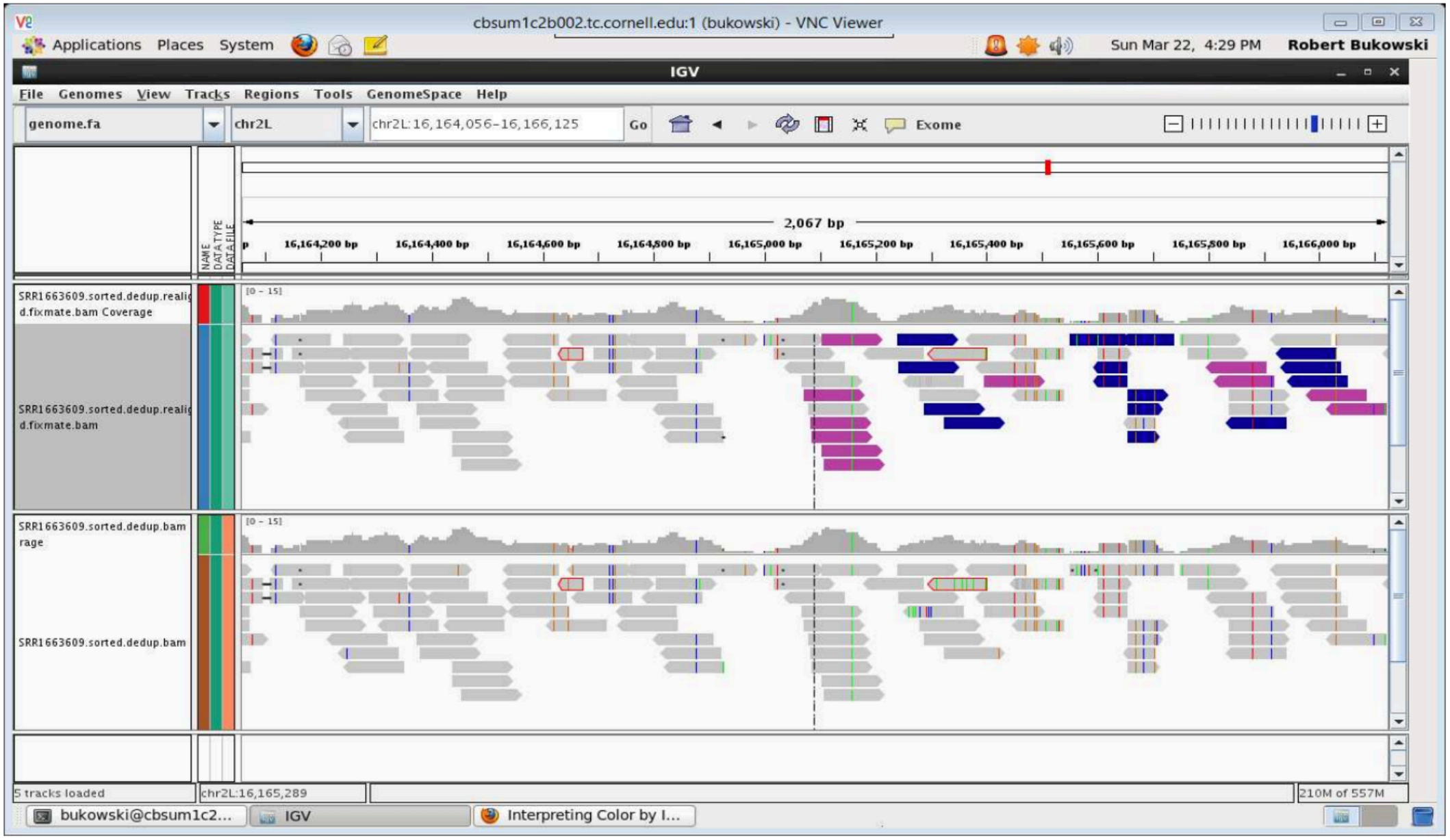
Visualizing read mapping/alignment: IGV

Look at multiple BAM files

Zoom in and out

Various color-coding schemes

Can load genome annotation track



VCF: Variant Call Format

Describes information about sequence variation.

BCF is the compressed version.

```
##fileformat=VCFv4.1
##fileDate=20180619
##source="Pilon version 1.22 Wed Mar 15 16:38:30 2017 -0400"
##PILON="--genome /Users/ukaraoz/Work/MTB/SFClusters/bin/data/ref/GCA_000277735.2_ASM27773v2_genomic.fna --bam test.sorted.duprem.bam --output out_pilon --variant"
##reference=file:/Users/ukaraoz/Work/MTB/SFClusters/bin/data/ref/GCA_000277735.2_ASM27773v2_genomic.fna
##contig=<ID=CP003248.2,length=4411709>
##FILTER=<ID=LowCov,Description="Low Coverage of good reads at location">
##FILTER=<ID=Amb,Description="Ambiguous evidence in haploid genome">
##FILTER=<ID=Del,Description="This base is in a deletion or change event from another record">
##INFO=<ID=DP,Number=1,Type=Integer,Description="Valid read depth; some reads may have been filtered">
##INFO=<ID=TD,Number=1,Type=Integer,Description="Total read depth including bad pairs">
##INFO=<ID=PC,Number=1,Type=Integer,Description="Physical coverage of valid inserts across locus">
##INFO=<ID=BQ,Number=1,Type=Integer,Description="Mean base quality at locus">
##INFO=<ID=MQ,Number=1,Type=Integer,Description="Mean read mapping quality at locus">
##INFO=<ID=QD,Number=1,Type=Integer,Description="Variant confidence/quality by depth">
##INFO=<ID=BC,Number=4,Type=Integer,Description="Count of As, Cs, Gs, Ts at locus">
##INFO=<ID=QP,Number=4,Type=Integer,Description="Percentage of As, Cs, Gs, Ts weighted by Q & MQ at locus">
##INFO=<ID=IC,Number=1,Type=Integer,Description="Number of reads with insertion here">
##INFO=<ID=DC,Number=1,Type=Integer,Description="Number of reads with deletion here">
##INFO=<ID=XC,Number=1,Type=Integer,Description="Number of reads clipped here">
##INFO=<ID=AC,Number=A,Type=Integer,Description="Allele count in genotypes, for each ALT allele, in the same order as listed">
##INFO=<ID=AF,Number=A,Type=Float,Description="Fraction of evidence in support of alternate allele(s)">
##INFO=<ID=SVTYPE,Number=1,Type=String,Description="Type of structural variant">
##INFO=<ID=SVLEN,Number=.,Type=String,Description="Difference in length between REF and ALT alleles">
##INFO=<ID=END,Number=1,Type=Integer,Description="End position of the variant described in this record">
##INFO=<ID=IMPRECISE,Number=0,Type=Flag,Description="Imprecise change from local reassembly (ALT contains Ns)">
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=AD,Number=.,Type=String,Description="Allelic depths for the ref and alt alleles in the order listed">
##FORMAT=<ID=DP,Number=1,Type=String,Description="Approximate read depth; some reads may have been filtered">
##ALT=<ID=DUP,Description="Possible segmental duplication">
#CHROM  POS ID  REF ALT QUAL    FILTER  INFO    FORMAT  SAMPLE
CP003248.2  1977    .   A   G   2357    PASS    DP=59;TD=59;BQ=40;MQ=60;QD=39;BC=0,0,59,0;QP=0,0,100,0;PC=90;IC=0;DC=0;XC=0;AC=2;AF=1.00    GT  1/1 1
CP003248.2  2532    .   T   C   2298    PASS    DP=57;TD=57;BQ=40;MQ=60;QD=40;BC=0,57,0,0;QP=0,100,0,0;PC=85;IC=0;DC=0;XC=0;AC=2;AF=1.00    GT  1/1 1
CP003248.2  4013    .   T   C   2629    PASS    DP=65;TD=65;BQ=40;MQ=60;QD=40;BC=0,65,0,0;QP=0,100,0,0;PC=95;IC=0;DC=0;XC=0;AC=2;AF=1.00    GT  1/1 1
CP003248.2  6112    .   G   C   2726    PASS    DP=69;TD=69;BQ=40;MQ=60;QD=39;BC=0,69,0,0;QP=0,100,0,0;PC=101;IC=0;DC=0;XC=0;AC=2;AF=1.00   GT  1/1 1
CP003248.2  6832    .   C   T   2427    PASS    DP=60;TD=60;BQ=40;MQ=60;QD=40;BC=0,0,0,60;QP=0,0,0,100;PC=88;IC=0;DC=0;XC=0;AC=2;AF=1.00    GT  1/1 1
CP003248.2  7297    .   A   G   2674    PASS    DP=70;TD=70;BQ=39;MQ=60;QD=38;BC=0,1,69,0;QP=0,0,100,0;PC=101;IC=0;DC=0;XC=0;AC=2;AF=1.00   GT  1/1 1
CP003248.2  24698   .   GCCGCGTTGCTCGGGGTAA G   .   PASS    DP=72;TD=73;BQ=40;MQ=59;QD=10;BC=0,19,0,0;QP=0,100,0,0;PC=106;IC=0;DC=53;XC=0;AC=2;AF=0.74  GT  1/1 1
CP003248.2  24716   .   A   G   371 Del;Amb DP=15;TD=68;BQ=183;MQ=12;QD=24;BC=9,0,5,1;QP=58,0,35,6;PC=105;IC=0;DC=53;XC=0;AC=1;AF=0.38  GT  0/1 1
CP003248.2  24720   .   CGTTGCTCGGGGTAACCGC C   .   PASS    SVTYPE=DEL;SVLEN=-18;END=24738  GT  1/1 1
CP003248.2  71581   .   C   CCGAGCGCTGTTCTGGCGCTAATCTGACGCTAGAATAG  2808    PASS    DP=71;TD=71;BQ=40;MQ=47;QD=39;BC=0,71,0,0;QP=0,100,0,0;PC=98;IC=17;DC=0;XC=0;AC=2;AF=0.30   GT  1/1 1
CP003248.2  150039  .   G   T   4   Amb;LowCov  DP=4;TD=4;BQ=40;MQ=60;QD=1;BC=0,0,3,1;QP=0,0,74,26;PC=28;IC=0;DC=0;XC=0;AC=1;AF=0.26    GT  0/1 1
CP003248.2  150893  .   G   GGGCCCGGCGGTCAGGGCGGTCAGGCGTTCGGCGCTAACAATATCGGCGGCACCGGCGGGGCCGGCGGCAACGGCGGGCCGGC .   PASS    SVTYPE=INS;SVLEN=181;END=150893;IMPRECISE   GT  1/1 1
CP003248.2  154189  .   A   G   844 Del DP=21;TD=41;BQ=40;MQ=51;QD=40;BC=0,0,21,0;QP=0,0,100,0;PC=86;IC=0;DC=0;XC=2;AC=2;AF=1.00    GT  1/1 1
CP003248.2  1618984 .   T   C   160 LowCov   DP=4;TD=4;BQ=40;MQ=60;QD=40;BC=0,4,0,0;QP=0,100,0,0;PC=108;IC=0;DC=0;XC=0;AC=2;AF=1.00  GT  1/1 1
CP003248.2  1637527 .   CCCCGCCGGG  C   .   Amb;LowCov  DP=5;TD=5;BQ=39;MQ=0;QD=31;BC=0,4,0,0;QP=0,100,0,0;PC=110;IC=0;DC=1;XC=0;AC=1;AF=0.20   GT  0/1 1
CP003248.2  3120174 .   T   C   1   Del;Amb;LowCov  DP=5;TD=23;BQ=41;MQ=3;QD=0;BC=0,1,0,4;QP=0,25,0,75;PC=102;IC=0;DC=0;XC=1;AC=1;AF=0.25   GT  0/1 1
CP003248.2  4386286 .   A   G   4633    Del;Amb DP=158;TD=161;BQ=40;MQ=60;QD=29;BC=69,0,89,0;QP=43,0,57,0;PC=200;IC=0;DC=0;XC=0;AC=1;AF=0.57    GT  0/1 1
```

FILTER

INFO

FORMAT

VARIANTS

VCF: Variant Call Format

#CHROM	POS	ID	REF	ALT	QUAL	FILTER	INFO	FORMAT	SAMPLE
CP003248.2	1977	.	A	G	2357	PASS	DP=59;TD=59;BQ=40;MQ=60;QD=39;BC=0,0,59,0;QP=0,0,100,0;PC=90;IC=0;DC=0;XC=0;AC=2;AF=1.00	GT	1/1

DP Valid read depth; some reads may have been filtered">
TD Total read depth including bad pairs">
PC Physical coverage of valid inserts across locus">
BQ Mean base quality at locus">
MQ Mean read mapping quality at locus">
QD Variant confidence/quality by depth">
BC Count of As, Cs, Gs, Ts at locus">
QP Percentage of As, Cs, Gs, Ts weighted by Q & MQ at locus">
IC Number of reads with insertion here">
DC Number of reads with deletion here">
XC Number of reads clipped here">
AC Allele count in genotypes, for each ALT allele, in the same order as listed">
AF Fraction of evidence in support of alternate allele(s)">
SVTYPE Type of structural variant">
SVLEN Difference in length between REF and ALT alleles">
END End position of the variant described in this record">
IMPRECISE Imprecise change from local reassembly (ALT contains Ns)

Take-homes

- Error mode of sequencing technology is key to alignment.
- Post-processing is key:
  - Duplicate marking
  - INDEL realignment