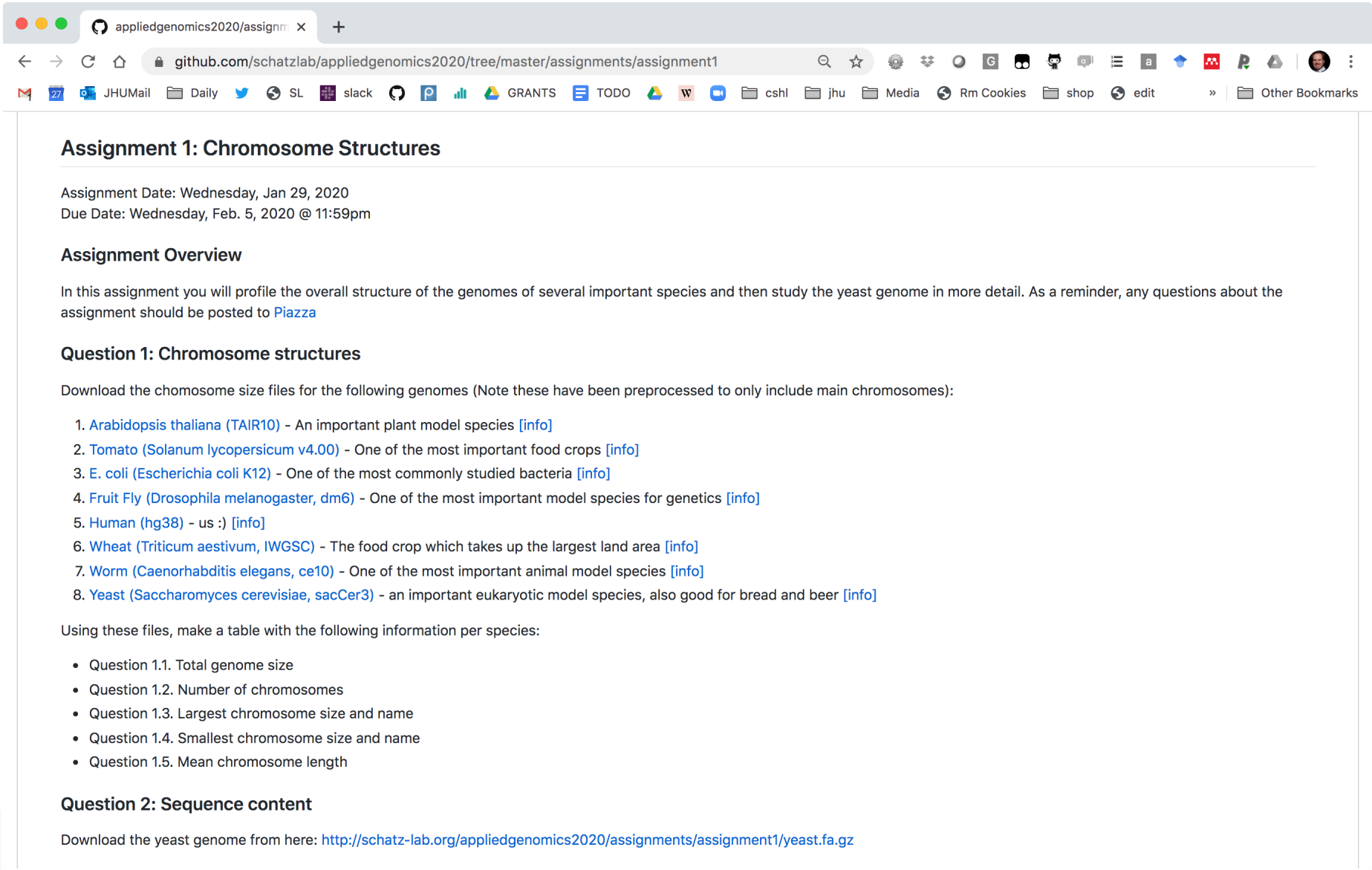
Assembly & Whole Genome Alignment

Michael Schatz

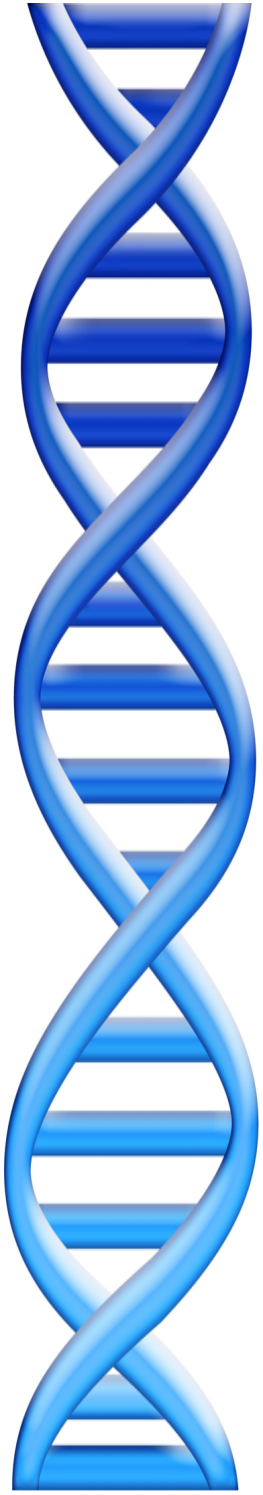
Feb 5, 2020

Lecture 4: Applied Comparative Genomics

Assignment 1: Chromosome Structures
Due Feb 5 @ 11:59pm



https://github.com/schatzlab/appliedgenomics2020



Outline

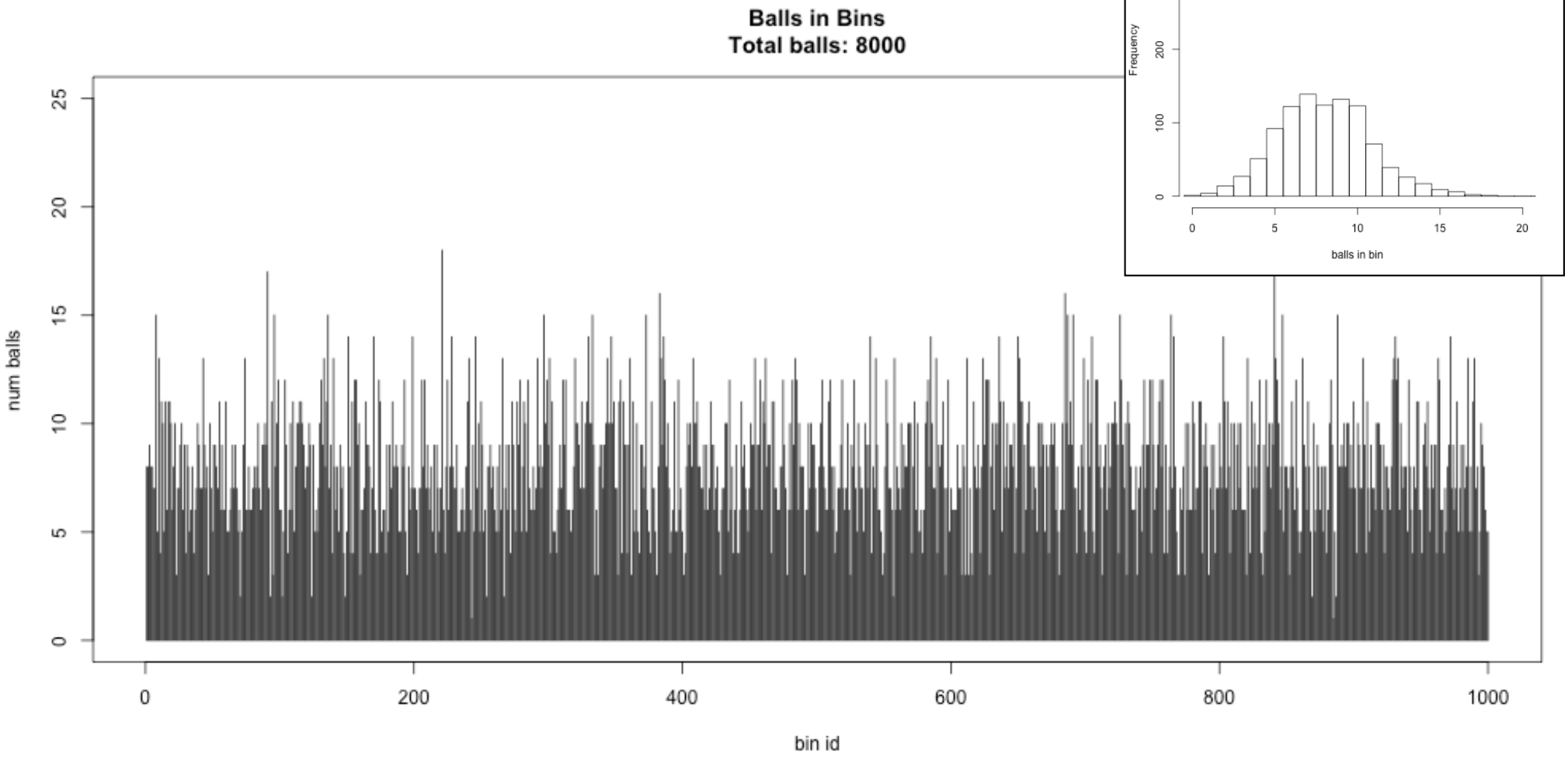
1. ***Assembly theory***
   - Assembly by analogy

2. ***Practical Issues***
   - Coverage, read length, errors, and repeats

3. ***Next-next-gen Assembly***
   - Canu: recommended for PacBio/ONT project

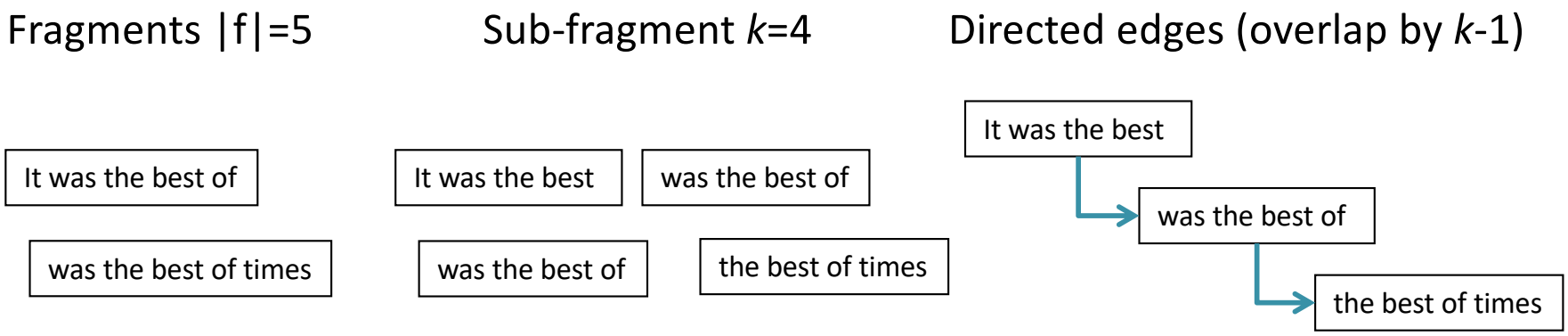
4. ***Whole Genome Alignment***
   - MUMmer recommended

8x sequencing



de Bruijn Graph Construction

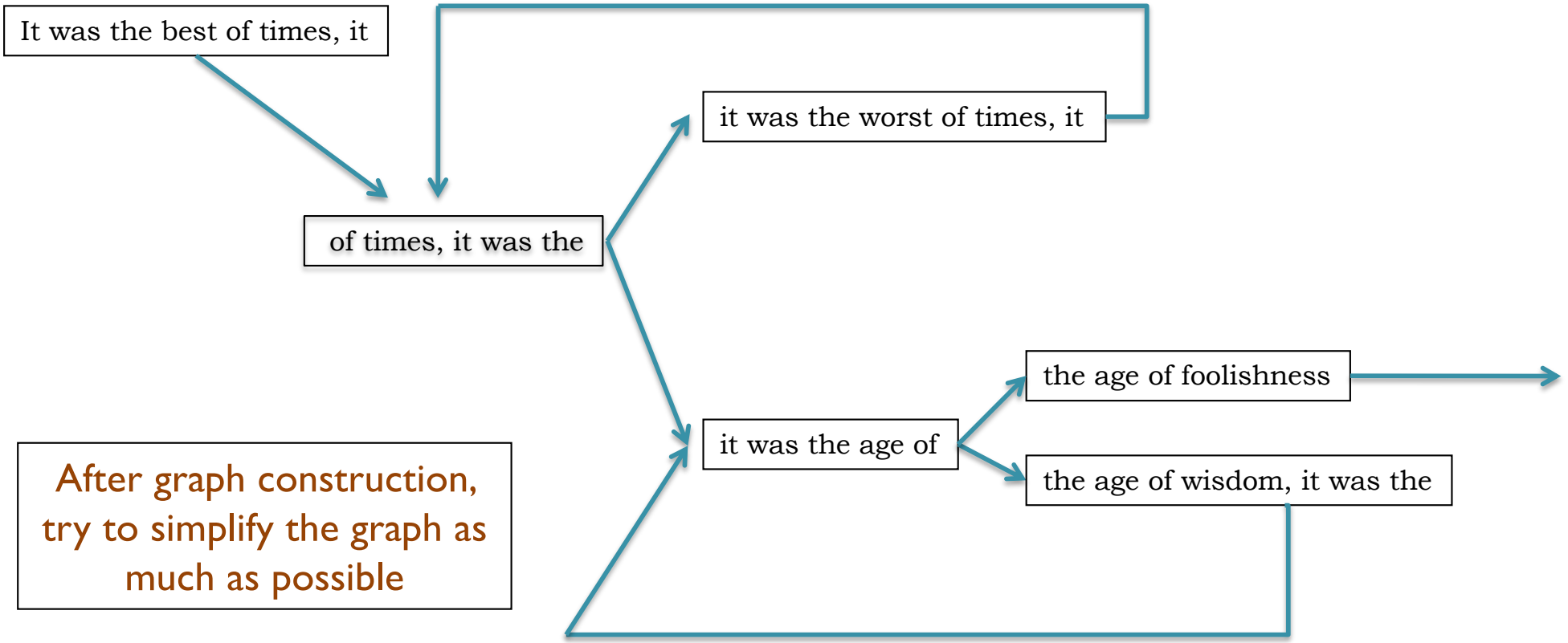
- $G_k = (V,E)$
  - V = Length-*k* sub-fragments
  - E = Directed edges between consecutive sub-fragments
    - Sub-fragments overlap by *k*-1 words



– Overlaps between fragments are implicitly computed

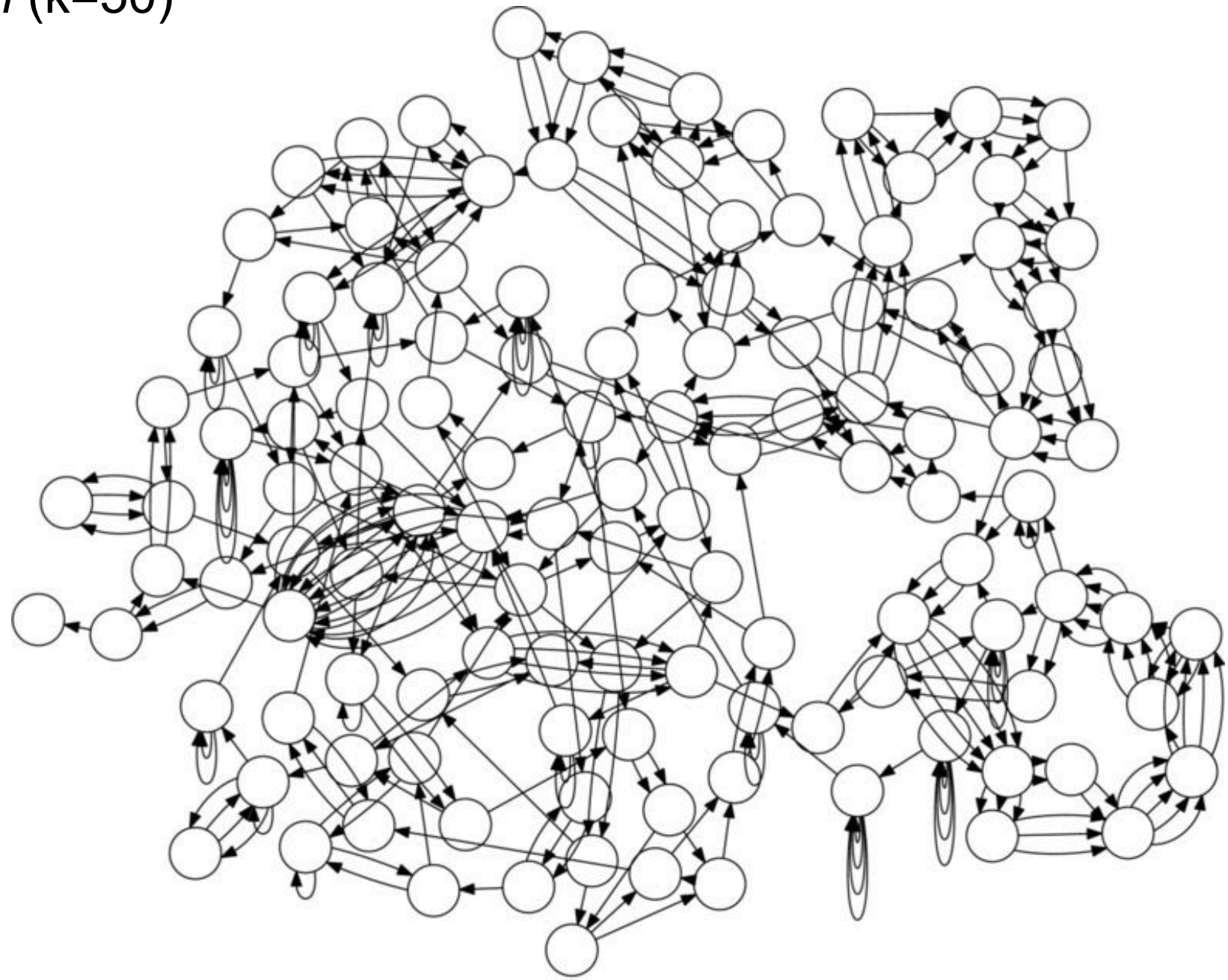
de Bruijn, 1946
Idury et al., 1995
Pevzner et al., 2001

de Bruijn Graph Assembly



After graph construction, try to simplify the graph as much as possible

E. coli (k=50)

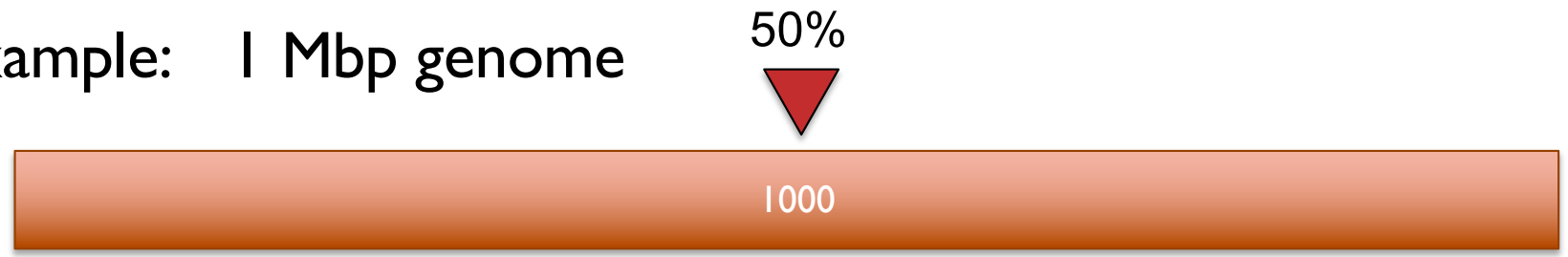


Reducing assembly complexity of microbial genomes with single-molecule sequencing
Koren et al (2013) Genome Biology. **14**:R101 https://doi.org/10.1186/gb-2013-14-9-r101

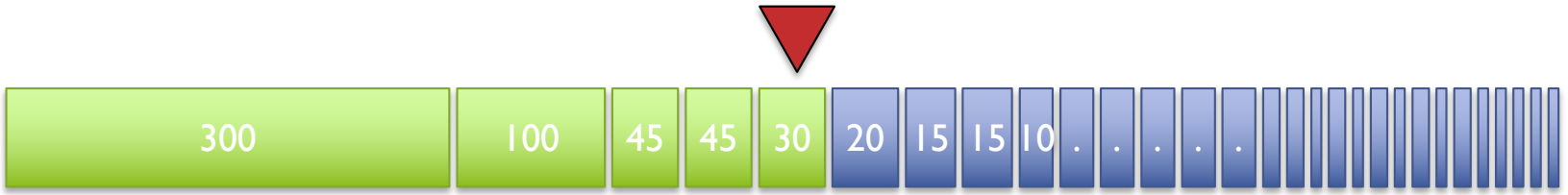
Contig N50

Def: 50% of the genome is in contigs as large as the N50 value

Example: 1 Mbp genome

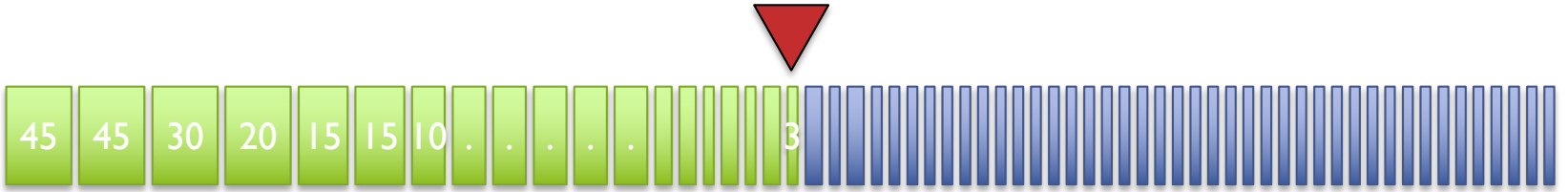


A



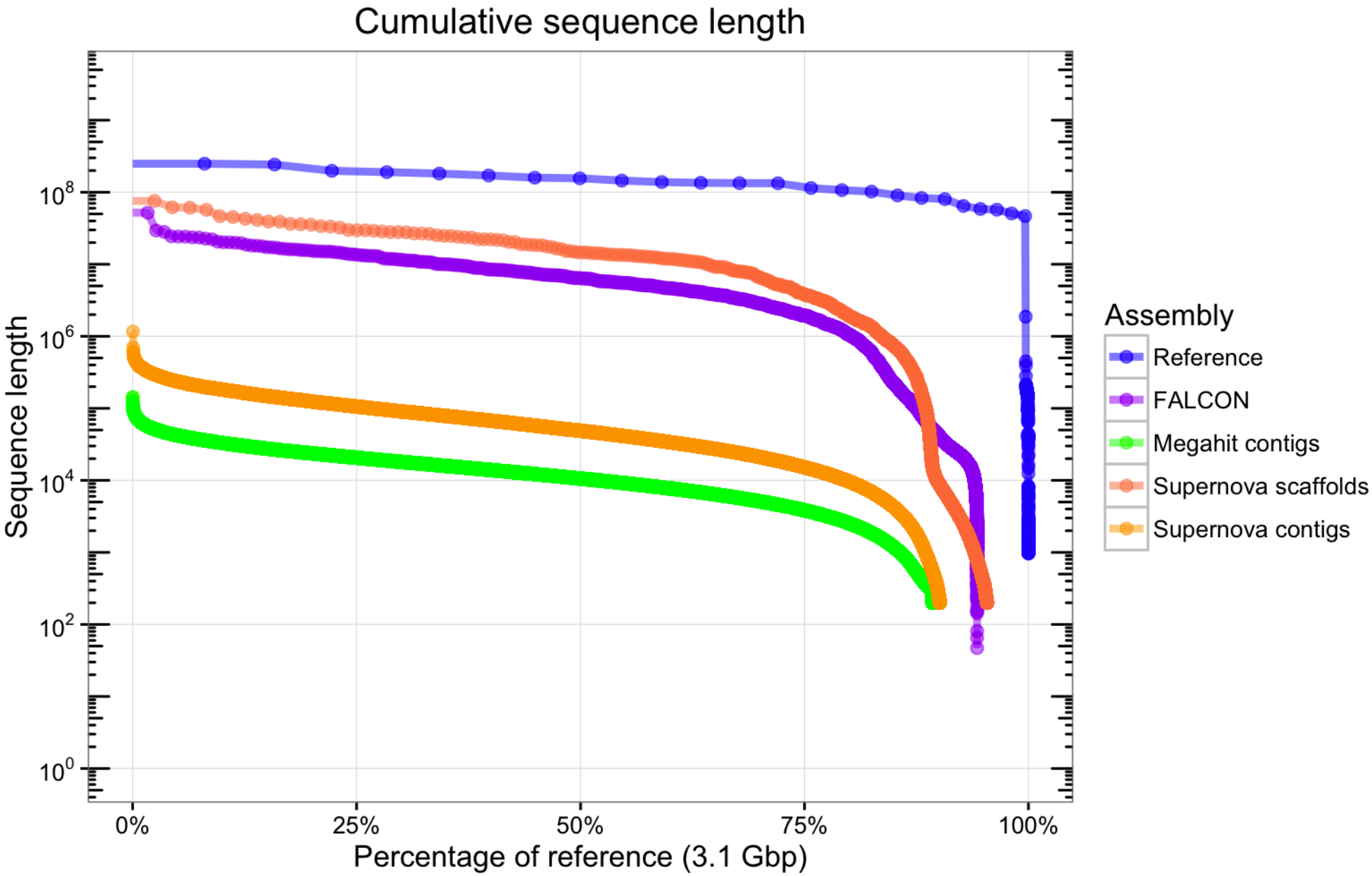
N50 size = 30 kbp

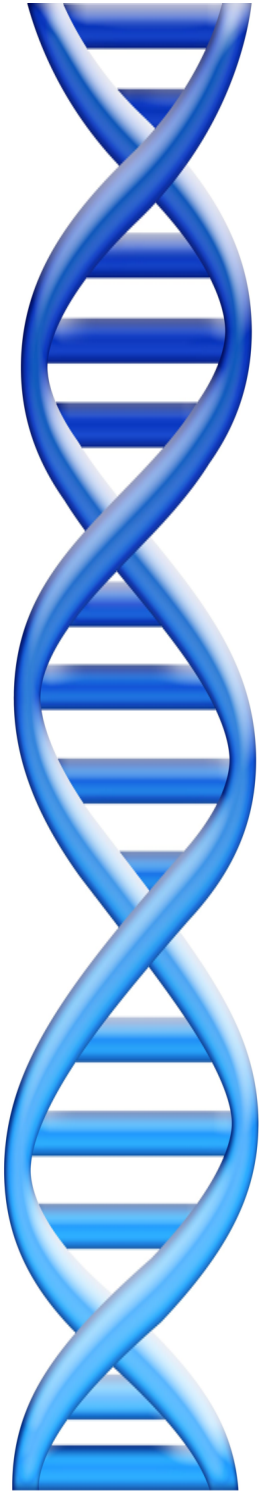
B



N50 size = 3 kbp

Contig Nchart





Outline

1. ***Assembly theory***
   - Assembly by analogy

2. ***Practical Issues***
   - Coverage, read length, errors, and repeats

3. ***Next-next-gen Assembly***
   - Canu: recommended for PacBio/ONT project

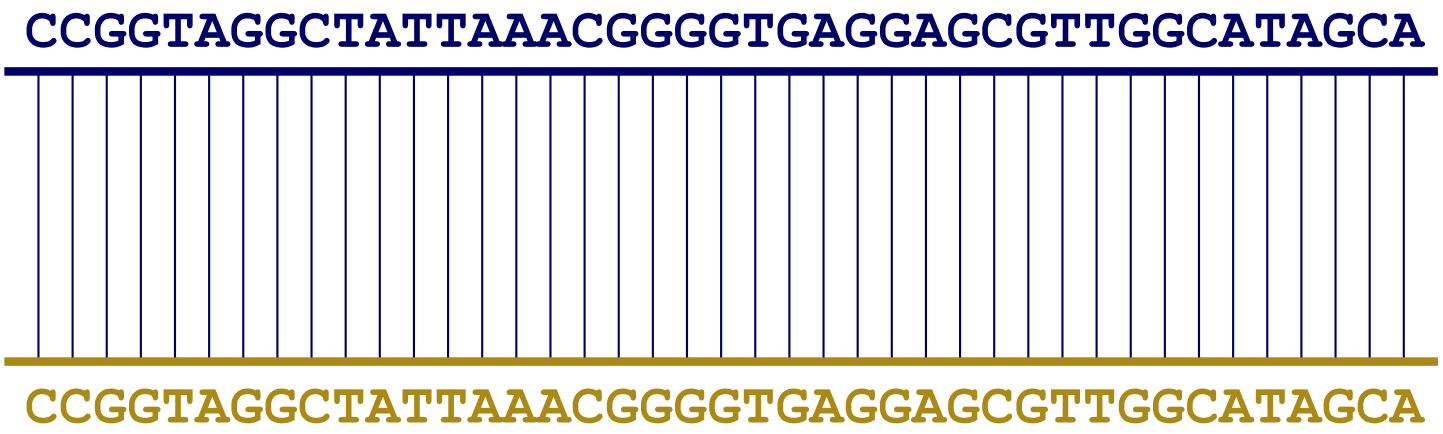
4. ***Whole Genome Alignment***
   - MUMmer recommended

Whole Genome Alignment with MUMmer

Slides Courtesy of Adam M. Phillippy
NHGRI

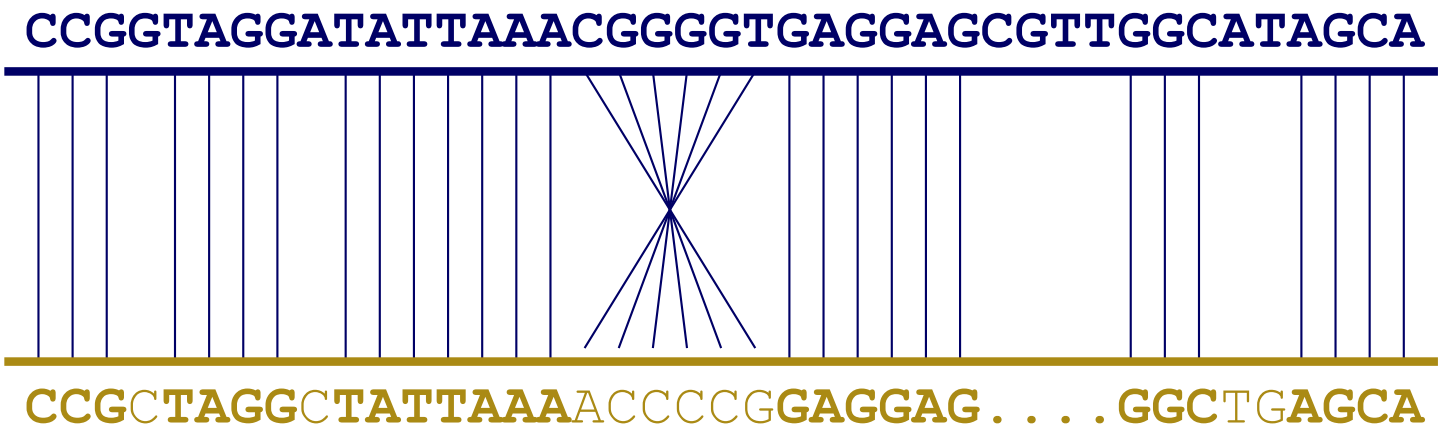
Goal of WGA

- For two genomes, *A* and *B*, find a mapping from each position in *A* to its corresponding position in *B*



Not so fast...

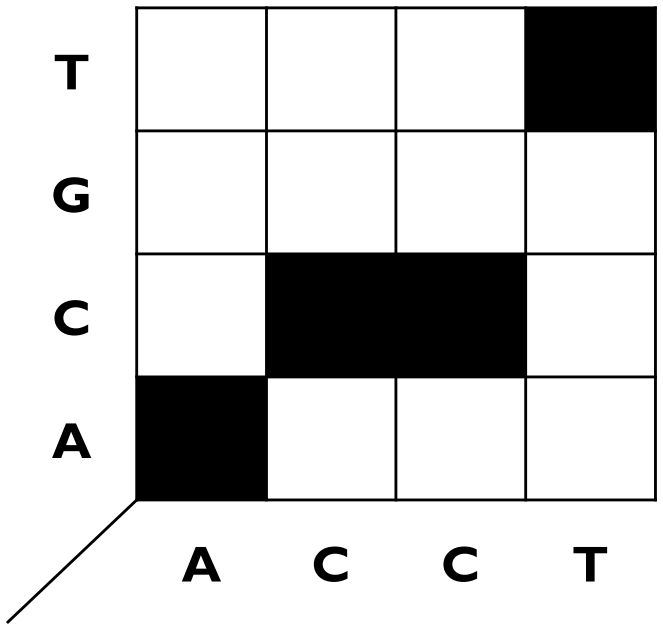
- Genome *A* may have insertions, deletions, translocations, inversions, duplications or SNPs with respect to *B* (sometimes all of the above)



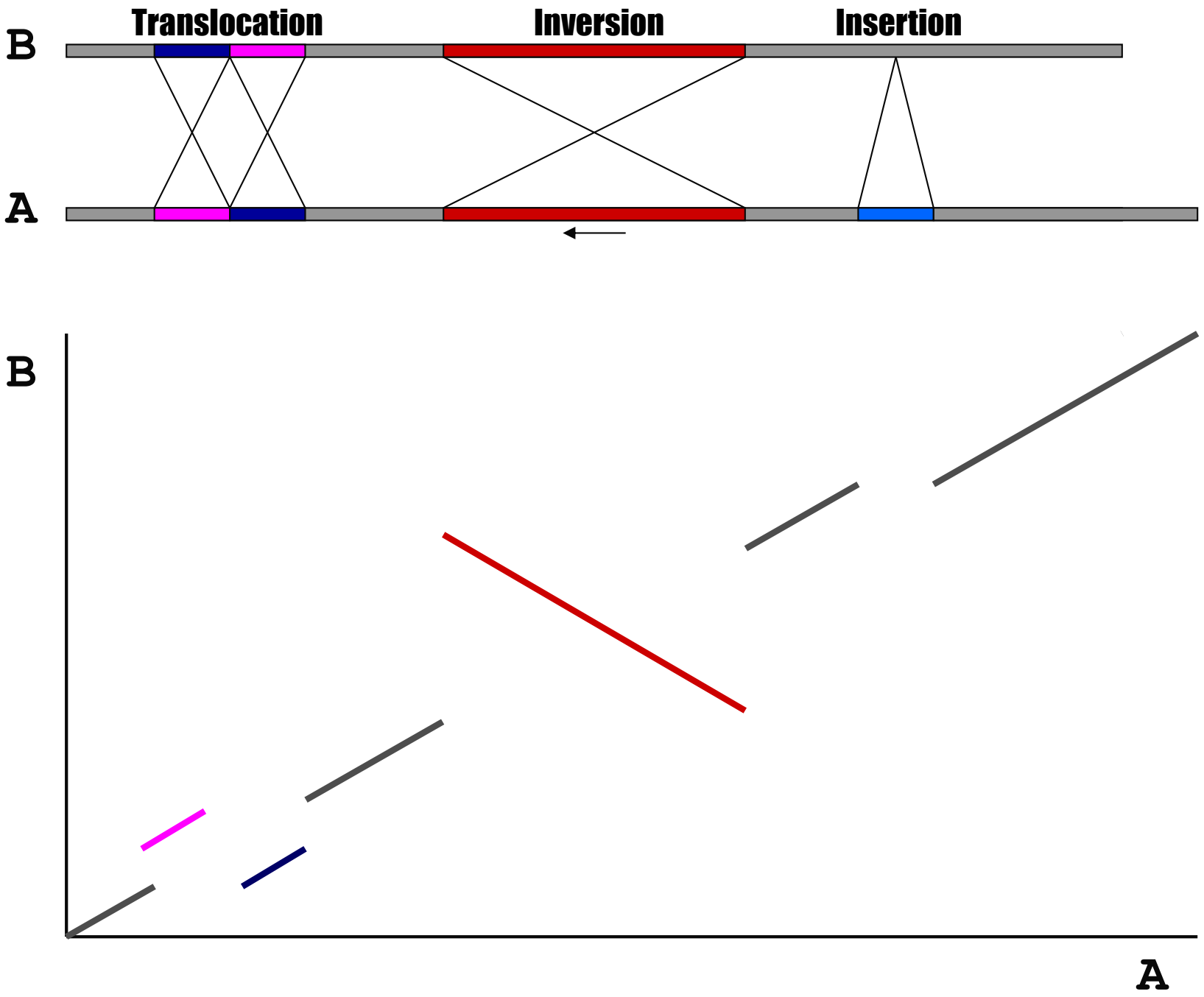
WGA visualization

- How can we visualize *whole* genome alignments?

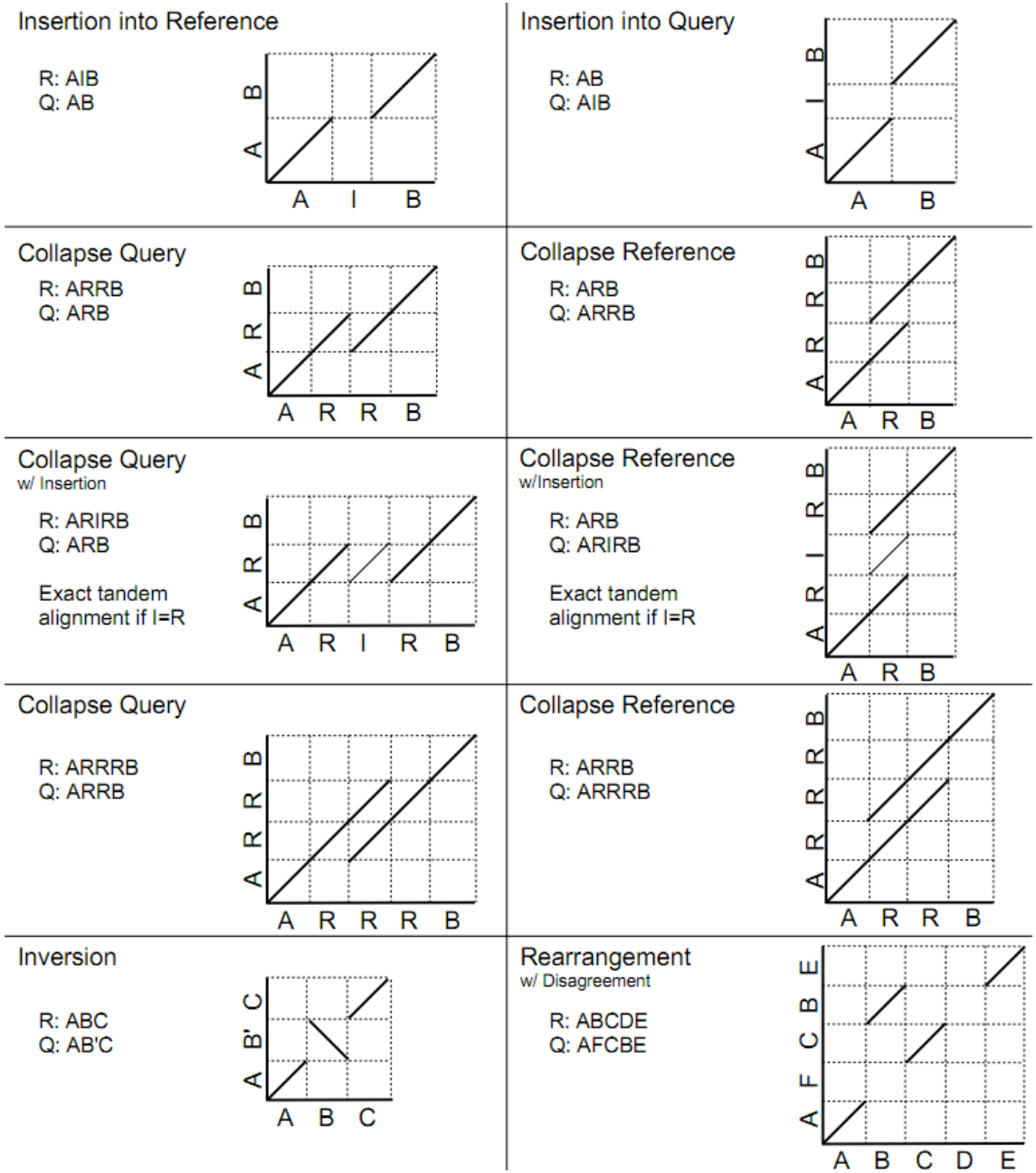
- With an alignment dot plot
  - *N* x *M* matrix
    - Let *i* = position in genome *A*
    - Let *j* = position in genome *B*
    - Fill cell *(i,j)* if A_i shows similarity to B_j



  - A perfect alignment between *A* and *B* would completely fill the positive diagonal

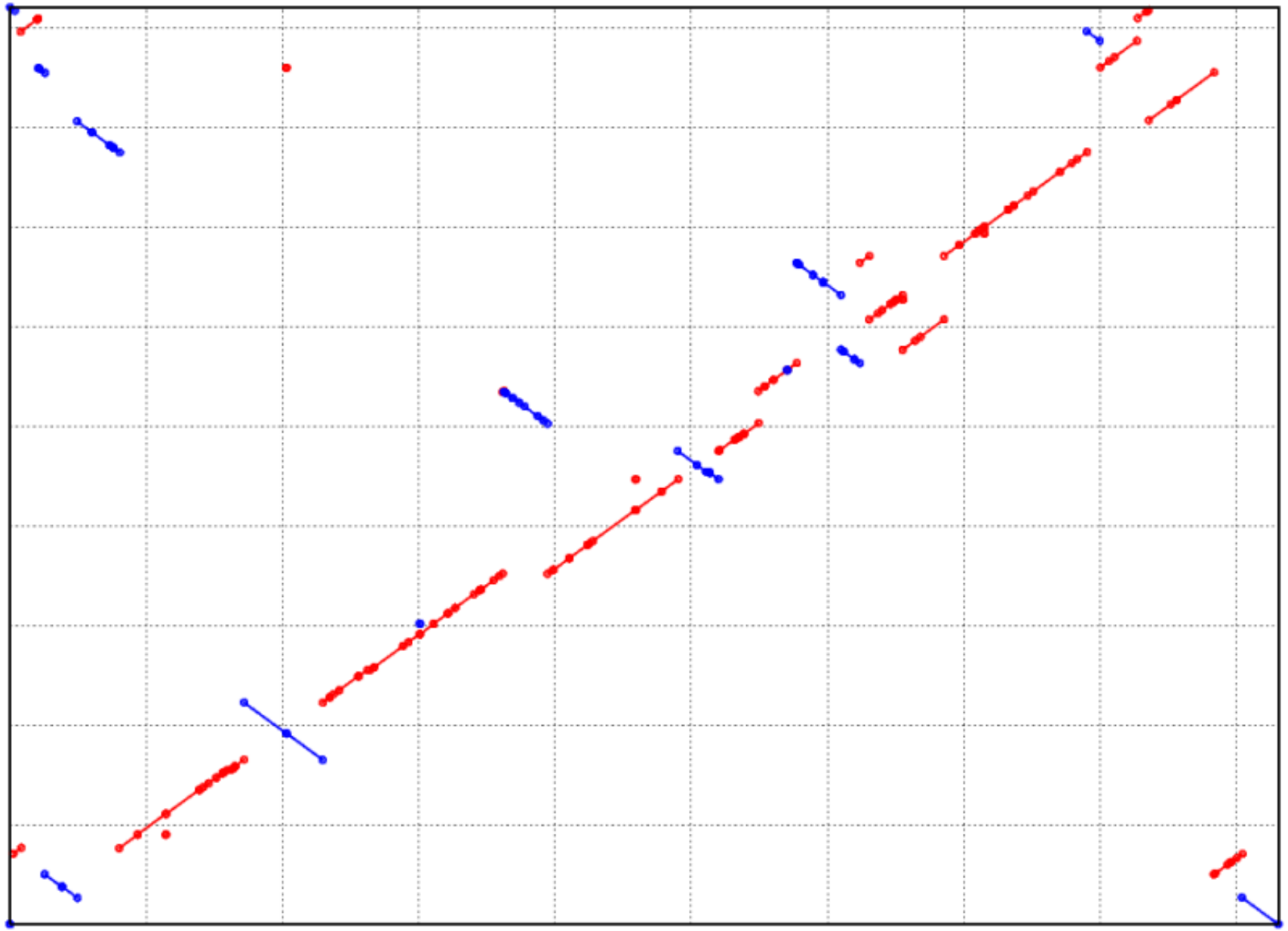

Translocation
Inversion
Insertion
B
A
B
A

SV Types



- Different structural variation types / misassemblies will be apparent by their pattern of breakpoints
- Most breakpoints will be at or near repeats
- Things quickly get complicated in real genomes

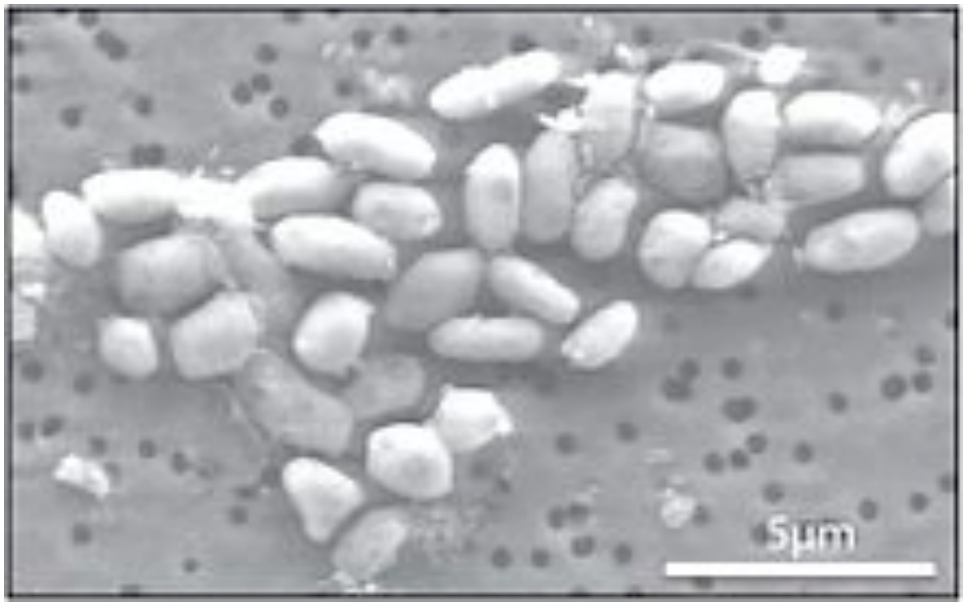
http://mummer.sf.net/manual/AlignmentTypes.pdf



Alignment of 2 strains of *Y. pestis*

http://mummer.sourceforge.net/manual/

Halomonas sp. GFAJ-1



Library 1: Fragment

Avg Read length: 100bp

Insert length: 180bp

Library 2: Short jump

Avg Read length: 50bp

Insert length: 2000bp

A Bacterium That Can Grow by Using Arsenic Instead of Phosphorus
Wolfe-Simon et al (2010) *Science. 332(6034)1163-1166.*

Digital Information Storage

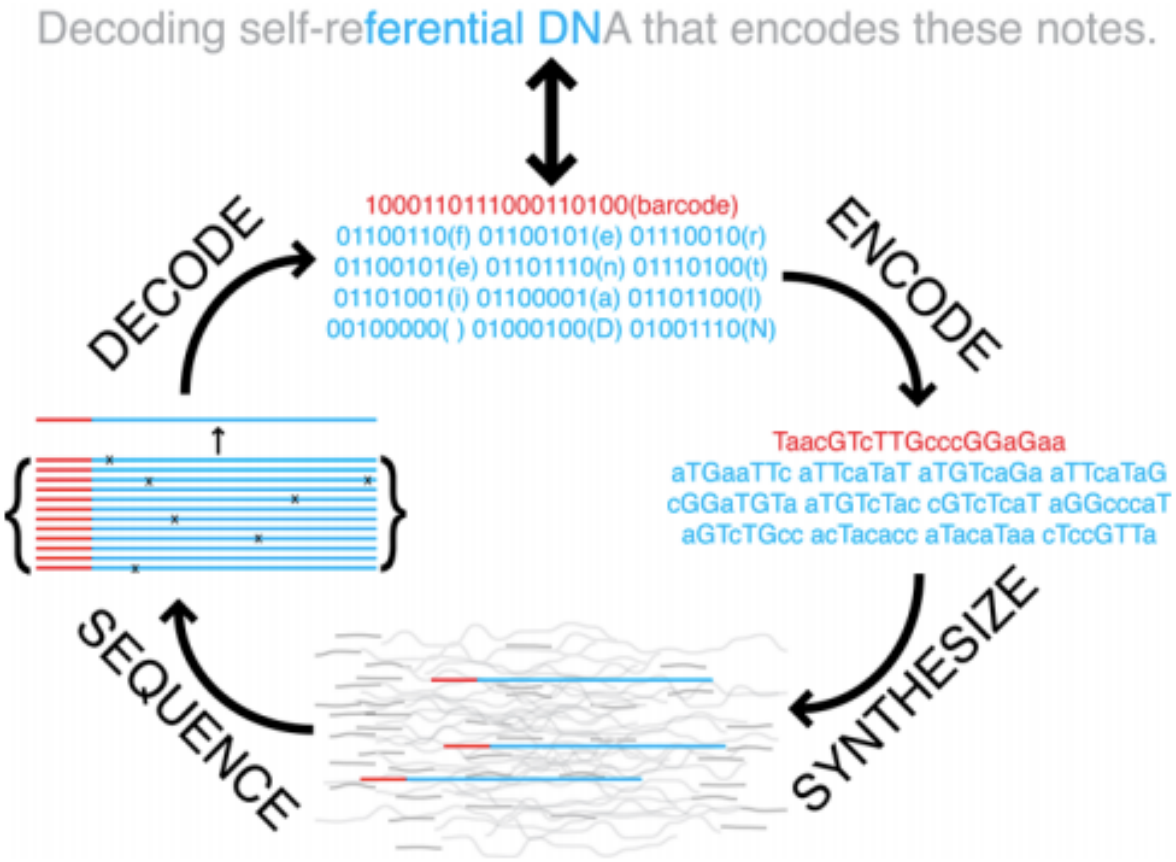


Fig. S1. Schematic of DNA information storage.

Encoding/decoding algorithm implemented in dna-encode.pl from David Dooling.

Next-generation Digital Information Storage in DNA
Church et al (2010) *Science*. 337(6102)1628

Assignment 2: Genome Assembly
Due Wednesday Feb 12 @ 11:59pm

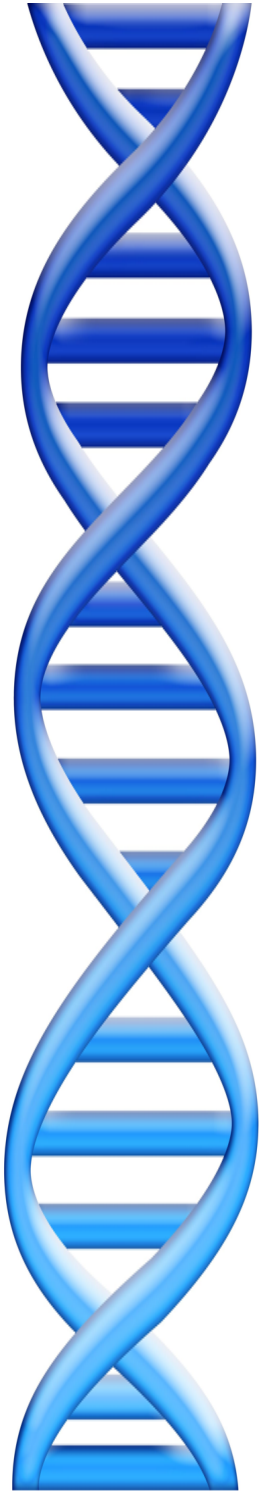
1. ***Setup Docker/Ubuntu***
2. ***Initialize Tools***
3. ***Download Reference Genome & Reads***
4. ***Decode the secret message***
    1. *Estimate coverage, check read quality*
    2. *Check kmer distribution*
    3. *Assemble the reads with spades*
    4. *Align to reference with MUMmer*
    5. *Extract foreign sequence*
    6. *dna-encode.pl -d*

https://github.com/schatzlab/appliedgenomics2020/blob/master/assignments/assignment2/README.md

Find and decode

```
nucmer -maxmatch ref.fasta \
   default/ASSEMBLIES/test/final.contigs.fasta
   -maxmatch     Find maximal exact matches (MEMs) without repeat filtering
   -p refctg     Set the output prefix for delta file

mummerplot --layout --png out.delta
   --layout      Sort the alignments along the diagonal
   --png         Create a png of the results

show-coords -rclo out.delta
   -r            Sort alignments by reference position
   -c            Show percent coverage
   -l            Show sequence lengths
   -o            Annotate each alignment with BEGIN/END/CONTAINS

samtools faidx default/ASSEMBLIES/test/final.contigs.fasta
   Index the fasta file

samtools faidx default/ASSEMBLIES/test/final.contigs.fasta \
    contig_XXX:YYY-ZZZ | ./dna-encode -d
```

See manual at http://mummer.sourceforge.net/manual



Outline

1. ***Assembly theory***
   - Assembly by analogy

2. ***Practical Issues***
   - Coverage, read length, errors, and repeats

3. ***Next-next-gen Assembly***
   - Canu: recommended for PacBio/ONT project

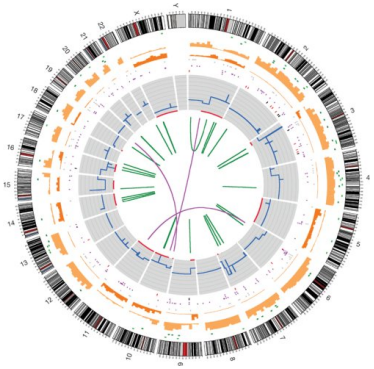
4. Whole Genome Alignment
   - MUMmer recommended

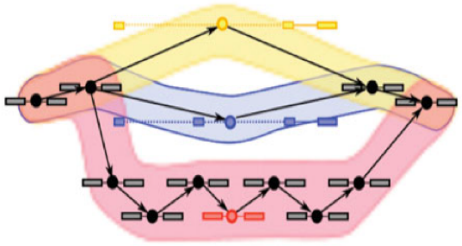
Assembly Applications

- Novel genomes

- Metagenomes

- Sequencing assays
  - Structural variations
  - Transcript assembly
  - …





Why are genomes hard to assemble?

1. ***Biological***:
   - (Very) High ploidy, heterozygosity, repeat content

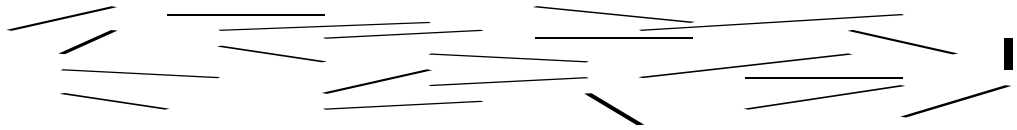
2. ***Sequencing***:
   - (Very) large genomes, imperfect sequencing

3. ***Computational***:
   - (Very) Large genomes, complex structure

4. ***Accuracy***:
   - (Very) Hard to assess correctness

Assembling a Genome

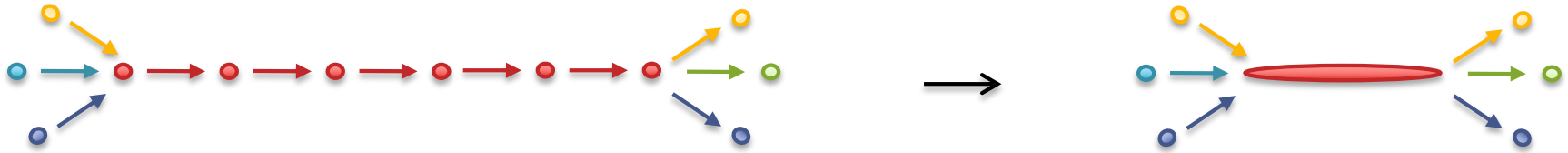
1. Shear & Sequence DNA



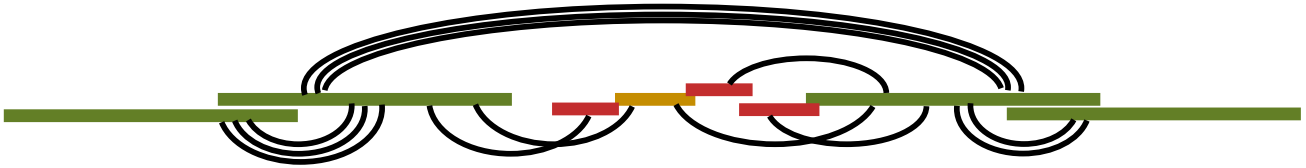
2. Construct assembly graph from reads (de Bruijn / overlap graph)

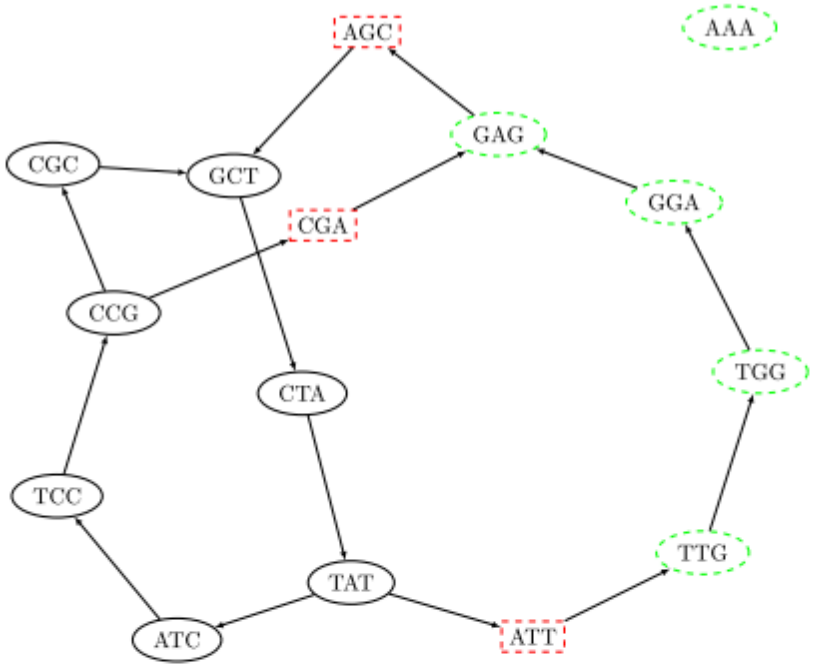
...AGCCTAGGGATGCGCGACACGT
GGATGCGCGACACGTCGCATATCCGGTTTGGTCAACCTCGGACGGAC
CAACCTCGGACGGACCTCAGCGAA...

3. Simplify assembly graph

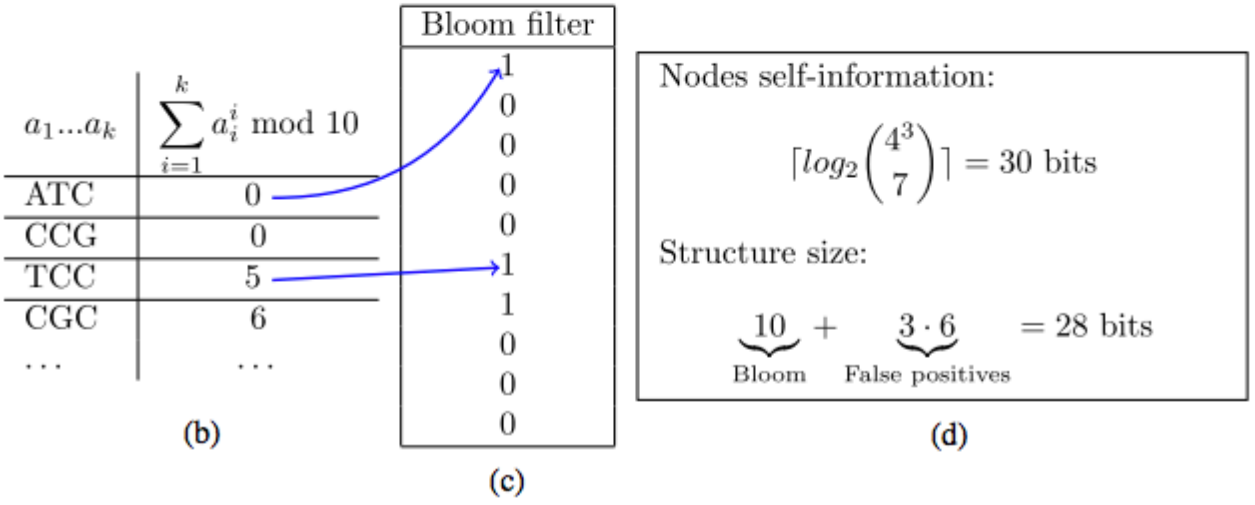


4. Detangle graph with long reads, mates, and other links





(a)



(b) (c) (d)

Table 2 de novo human genome (NA18507) assemblies

Method	Minia	C. & B.	ABySS	SOAPdenovo
Value of *k* chosen	27	27	27	25
Number of contigs (M)	3.49	7.69	4.35	-
Longest contig (kbp)	18.6	22.0	15.9	-
Contig N50 (bp)	1156	250	870	886
Sum (Gbp)	2.09	1.72	2.10	2.08
Nb of nodes/cores	1/1	1/8	21/168	1/16
Time (wall-clock, h)	23	50	15	33
Memory (sum of nodes, GB)	**5.7**	**32**	**336**	**140**

de novo human genome (NA18507) assemblies reported by our assembler (Minia), Conway and Bromage assembler [9], ABySS [8], and SOAPdenovo [7]. Contigs shorter than 100 bp were discarded. Assemblies were made without any pairing information.

Space-efficient and exact de Bruijn graph representation based on a Bloom filter

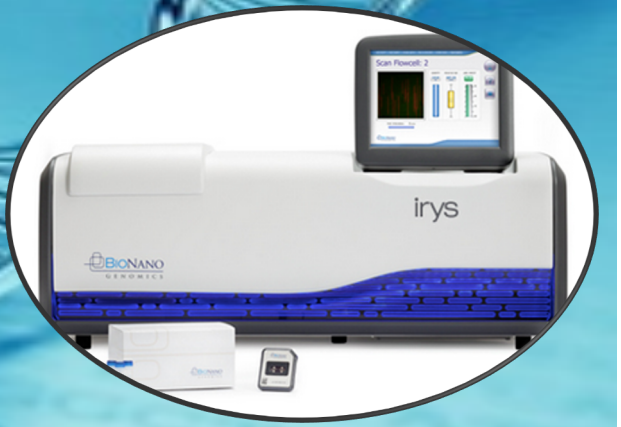
Chikhi and Rizk (2013) *Algorithms for Molecular Biology*. 8:22

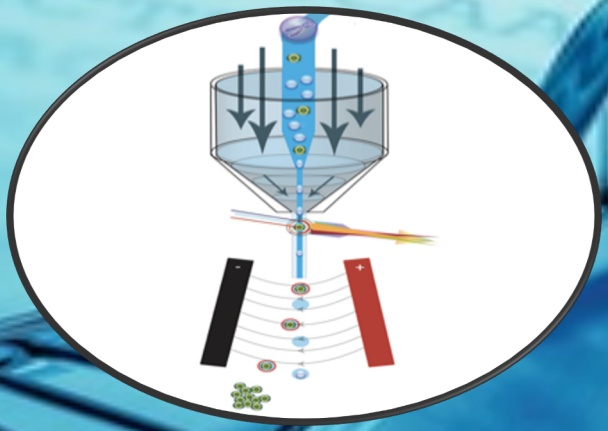
Genomics Arsenal in the year 2020

Sample Preparation

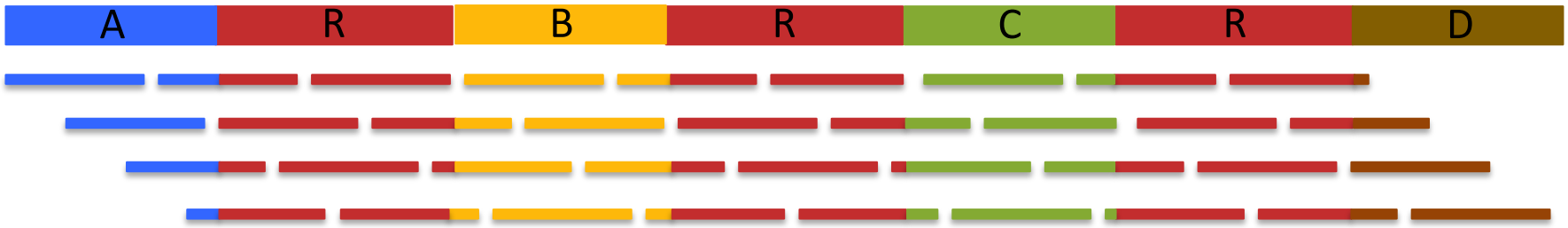
Sequencing

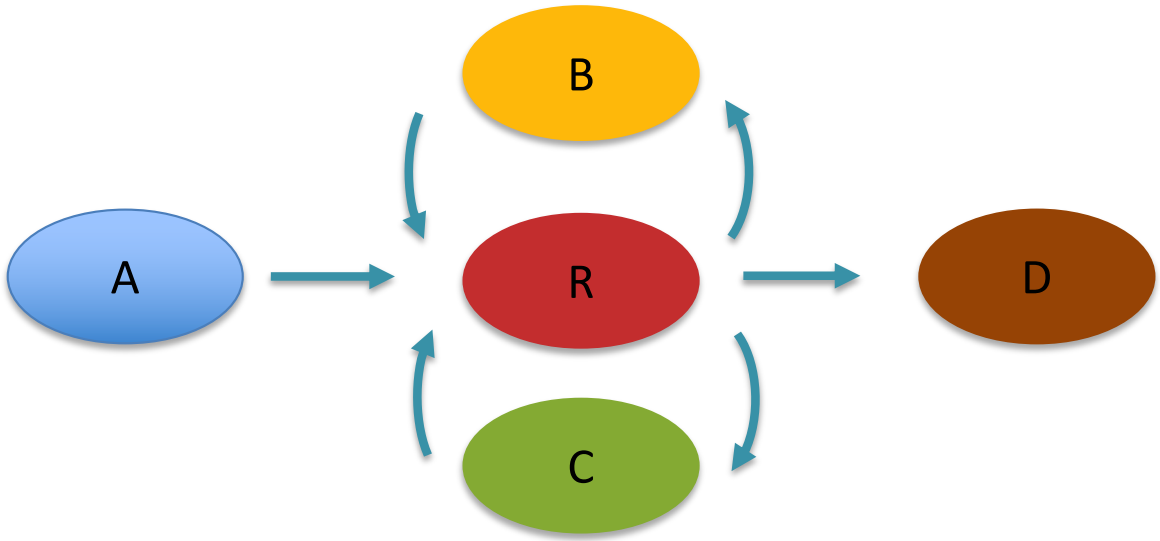
Chromosome Mapping



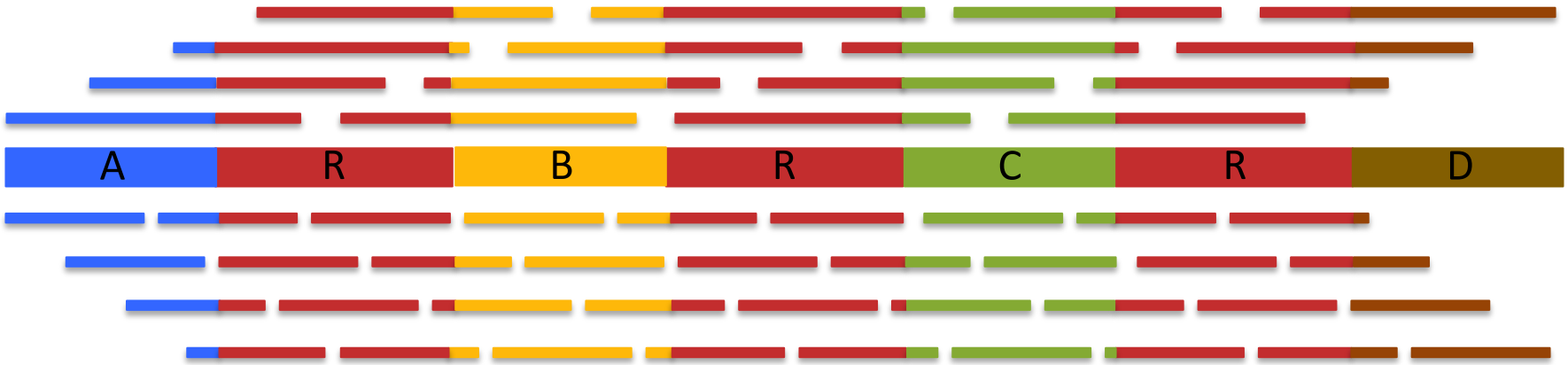


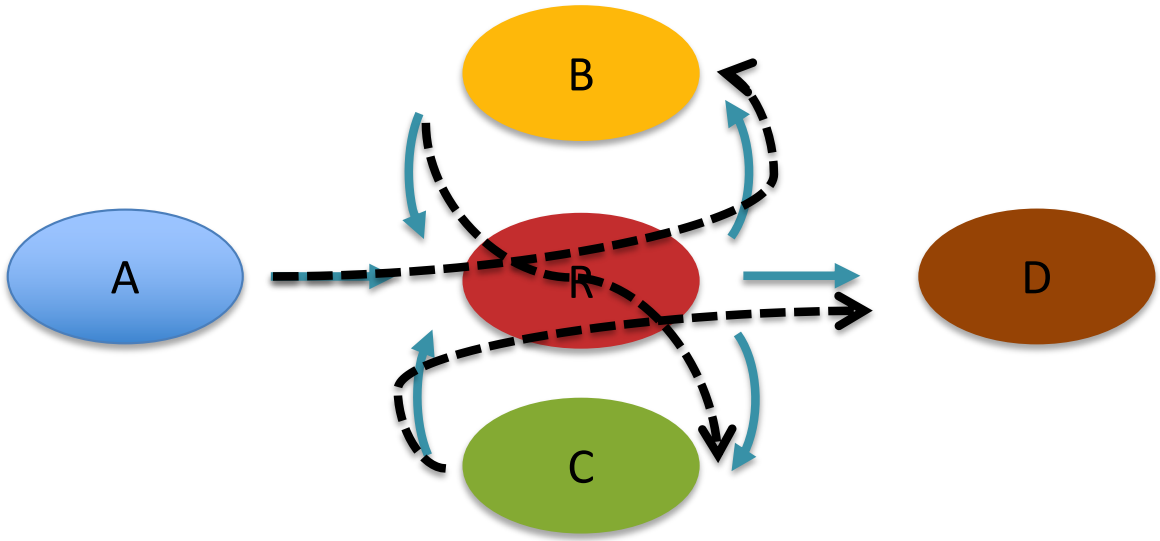
Assembly Complexity



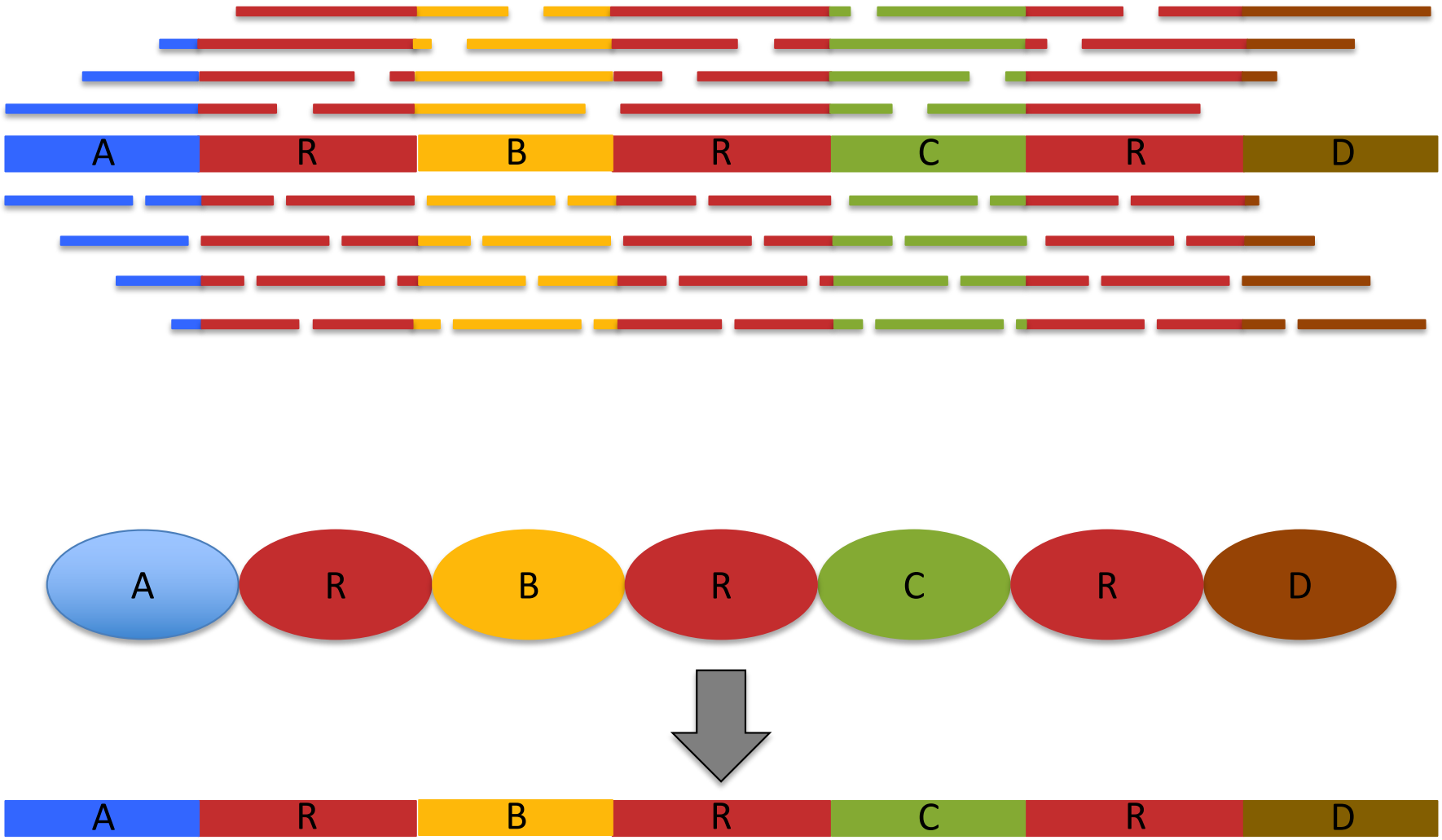


Assembly Complexity





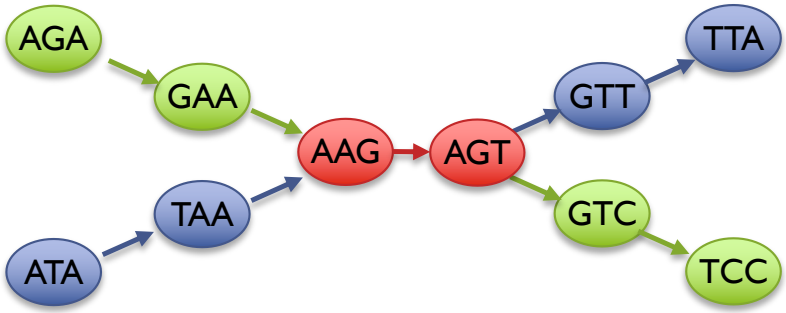
Assembly Complexity



The advantages of SMRT sequencing
Roberts, RJ, Carneiro, MO, Schatz, MC (2013) *Genome Biology*. 14:405

Two Paradigms for Assembly

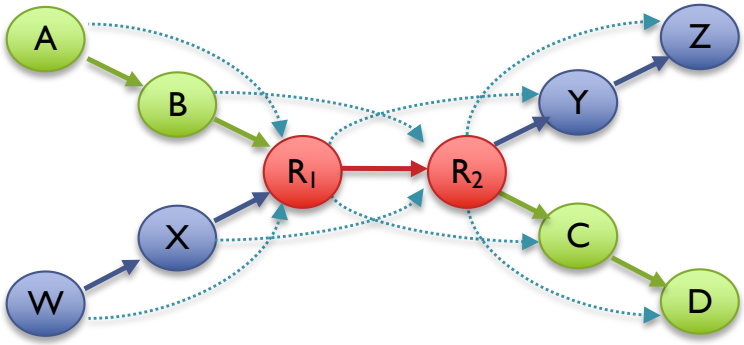
de Bruijn Graph



Short read assemblers

- Repeats depends on word length
- Read coherency, placements lost
- Robust to high coverage

Overlap Graph

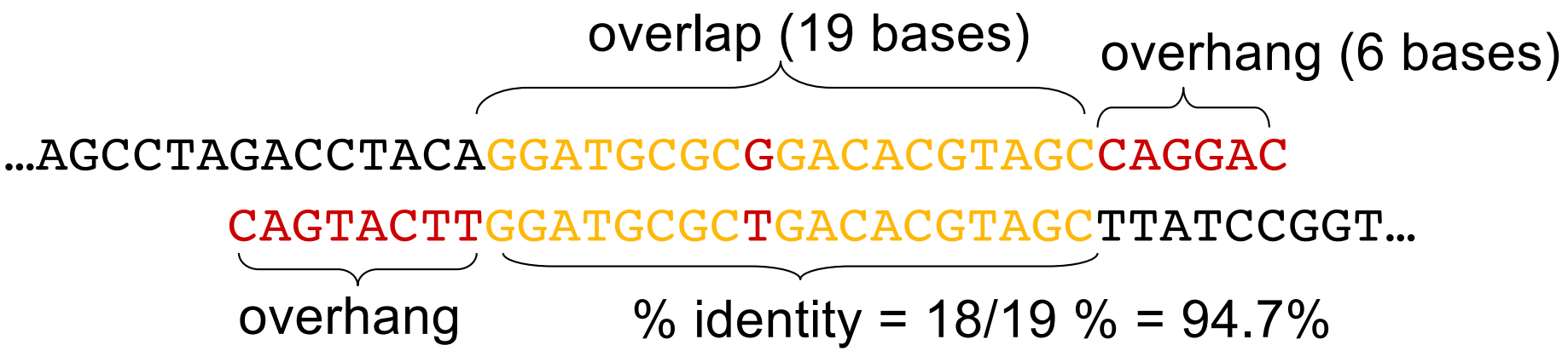


Long read assemblers

- Repeats depends on read length
- Read coherency, placements kept
- Tangled by high coverage

Assembly of Large Genomes using Second Generation Sequencing
Schatz MC, Delcher AL, Salzberg SL (2010) *Genome Research*. 20:1165-1173.

Overlap between two sequences



overlap - region of similarity between regions
overhang - un-aligned ends of the sequences

The assembler screens merges based on:

- length of overlap
- % identity in overlap region
- maximum overhang size.

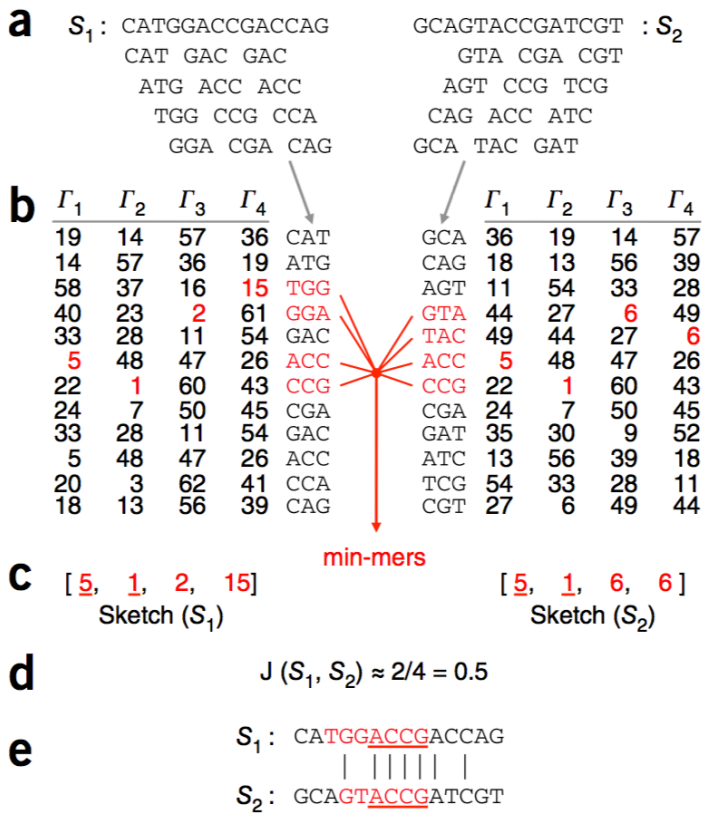
[How do we compute the overlap?]

[Do we really want to do all-vs-all?]

Very fast approximate overlapping

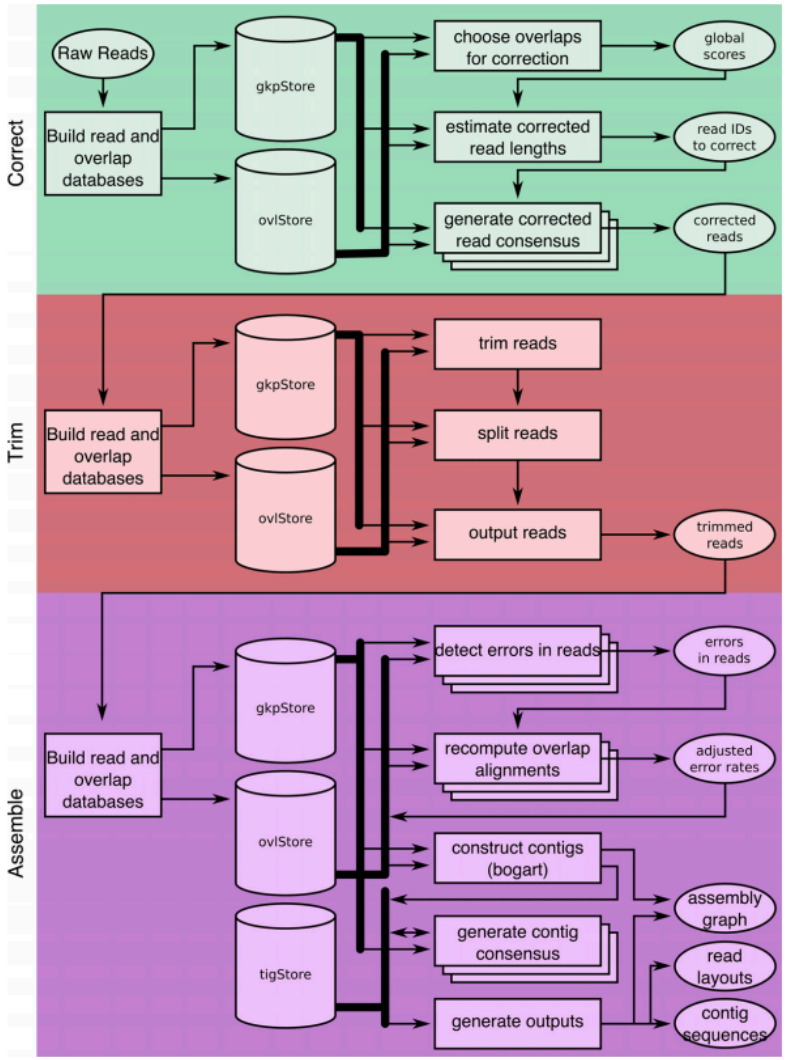
Maybe we don't need to compute the exact identity of the overlap region, just approximate it

- If two reads overlap, they should share many of the same kmers: Their Jaccard coefficient should be high: |intersection| / |union|
- But tracking all of the kmers for a read is a lot of overhead
- Instead, compare the "sketch" of the reads: a small fraction of kmers carefully chosen
- LSH: Find the sketch by applying N hash functions to the kmers, and keeping the minimum hash values reported from each (N=4 in example)
- This forms a nice "random" sample of the reads, and the Jaccard coefficient is a good approximation of the sequence similarity



Assembling large genomes with single-molecule sequencing and locality-sensitive hashing
Berlin et al (2015) *Nature Biotechnology*

Canu Workflow



Three rounds of analysis:

1. ***Error Correction:*** Use MHAP to overlap the reads, then compute a mini assembly centered around each read of good overlaps to error correct

2. ***Trim:*** Use MHAP to recompute overlaps to find regions that are not well supported and discard

3. ***Unitigging:*** Use Dynamic Programming to carefully overlap the error corrected reads, construct overlap graph, and then "unitig" those overlaps to build the contigs

Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation
Koren et al (2017) *Genome Research*

Unitigging: Pruning the Overlap Graph

The overlap graph has many redundant edges:

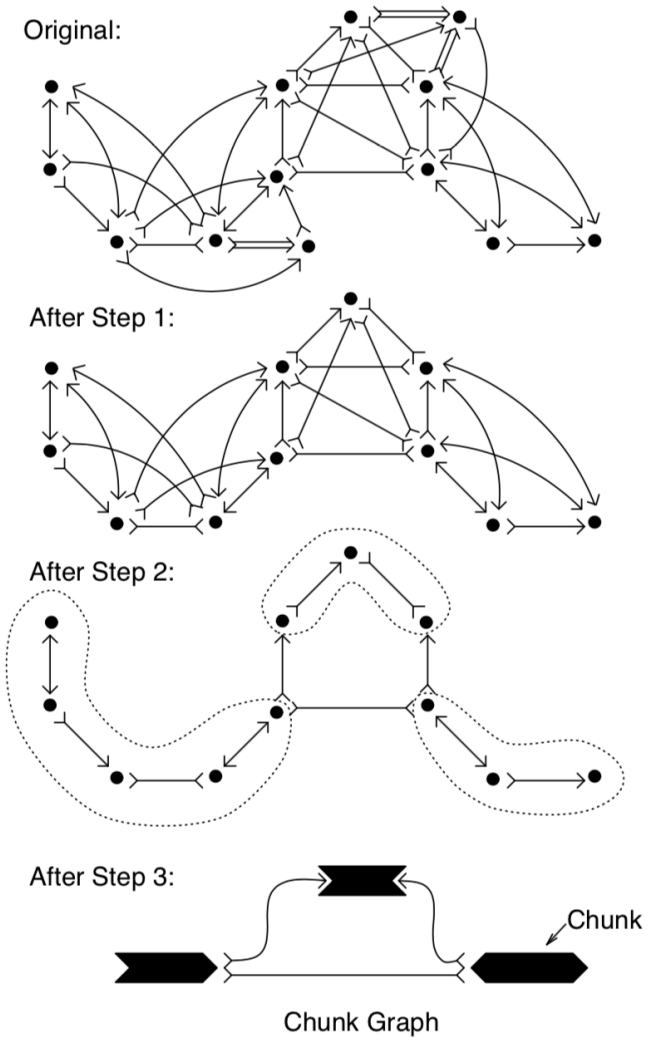
- If the average coverage is D, we should expect D overlaps at the beginning of the read, and D at the end

Transform the graph to simplify the assembly problem (without changing the valid solutions):

1. ***Contained reads removal:*** Short reads that are substrings of longer reads don't advance the assembly, remove those nodes and all of the edges

2. ***Transitive edge removal:*** If A -> B, and B->C, remove the transitive edge A->C

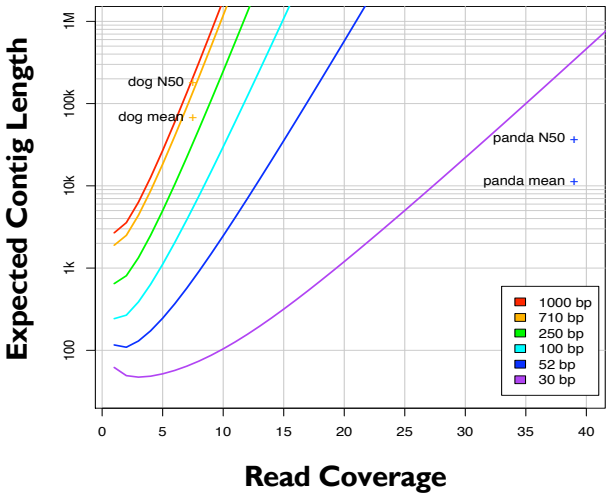
3. ***"Chunkification":*** Linear subgraphs define uniquely assemblible segments: "unitigs"



Towards Simplifying and Accurately Formulating Fragment Assembly
Myers (1995) *J Comput Biol.* Summer;2(2):275-90.

Ingredients for a good assembly

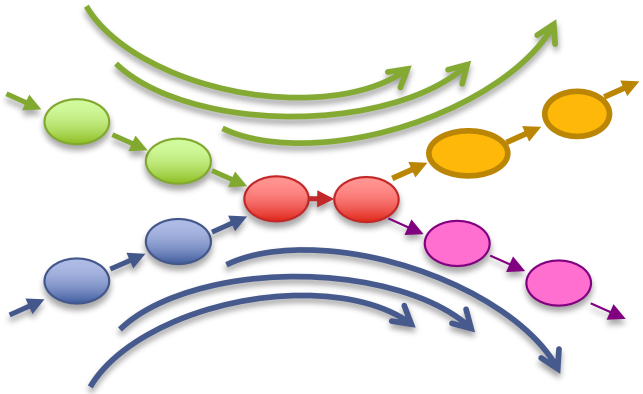
Coverage



High coverage is required

- Oversample the genome to ensure every base is sequenced with long overlaps between reads
- Biased coverage will also fragment assembly

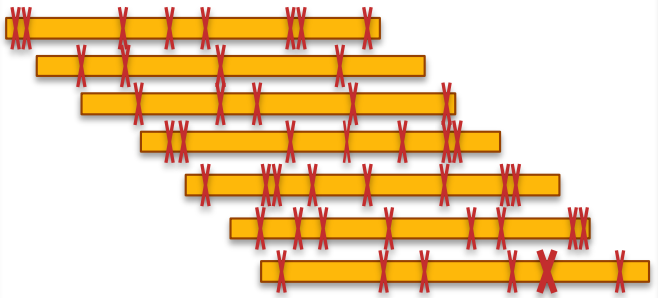
Read Length



Reads & mates must be longer than the repeats

- Short reads will have ***false overlaps*** forming hairball assembly graphs
- With long enough reads, assemble entire chromosomes into contigs

Quality



Errors obscure overlaps

- Reads are assembled by finding kmers shared in pair of reads
- High error rate requires very short seeds, increasing complexity and forming assembly hairballs

Current challenges in *de novo* plant genome sequencing and assembly
Schatz MC, Witkowski, McCombie, WR (2012) *Genome Biology*. 12:243

Coverage Statistics

$$\text{sequencing\_coverage} = \frac{\text{total\_bases\_sequenced}}{\text{genome\_size}}$$

$$\text{genome\_size} = \frac{\text{total\_bases\_sequenced}}{\text{sequencing\_coverage}}$$

$$\text{genome\_size} = \frac{100\text{Gb}}{50\text{x}} = 2\text{Gb}$$

But how can you figure out
the coverage without a genome?

K-mer counting

Kmer-ize

```
Read 1: GATTACA  => GAT,ATT,TTA,TAC,ACA
Read 2: TACAGAG  => TAC,ACA,CAG,AGA,GAG
Read 3: TTACAGA  => TTA,TAC,ACA,CAG,AGA
```

list

```
GAT   ACA   ACA:3
ATT   ACA
TTA   ACA
TAC   AGA   AGA:2
ACA   AGA
TAC   ATT   ATT:1
ACA   CAG   CAG:2
CAG   CAG
AGA   GAG   GAG:1
GAG   GAT   GAT:1
TTA   TAC   TAC:3
TAC   TAC
ACA   TAC
CAG   TTA   TTA:2
AGA   TTA
```

sort count

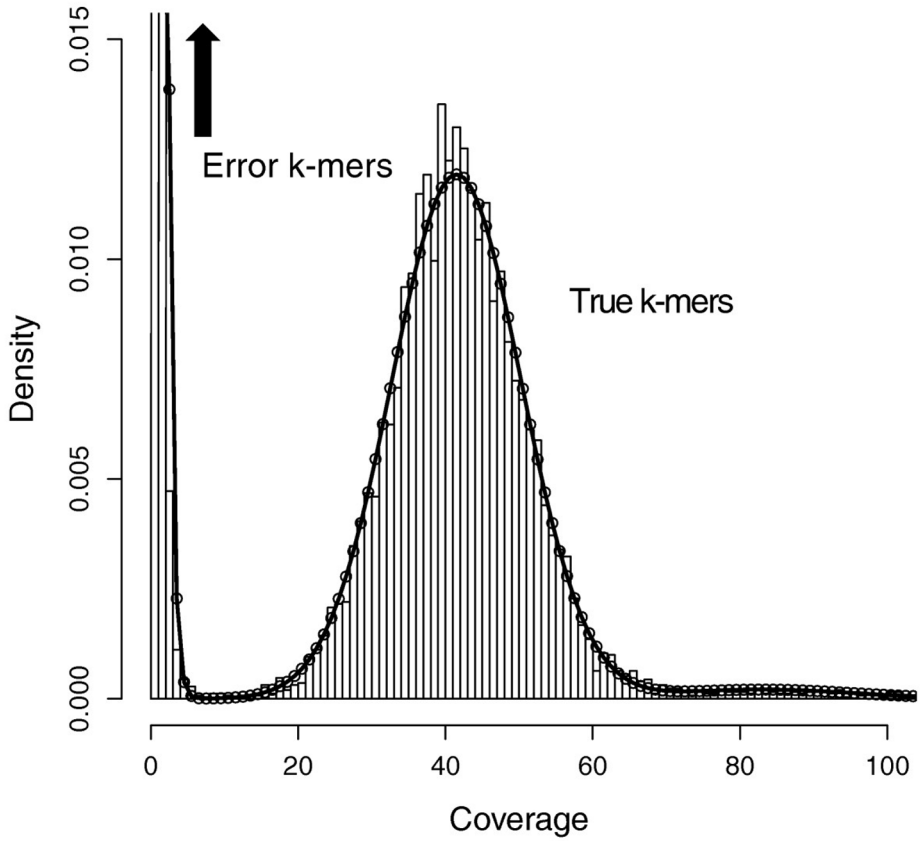
tally

```
3 kmers occur 1x
3 kmers occur 2x
2 kmers occur 3x
```

From read k-mers alone, can learn something about how frequently different sequences occur (aka coverage)

Fast to compute even over huge datasets

K-mer counting in real genomes



- The tally of k-mer counts in real genomes reveals the coverage distribution.
- Here we sequenced 120Gb of reads from a female human (haploid human genome size is 3Gb), and indeed we see a clear peak centered at 40x coverage
- There are also many kmers that only occur <5 times. These are from errors in the reads
- There are also kmers that occur many times (>>70 times). These are repeats in the genome

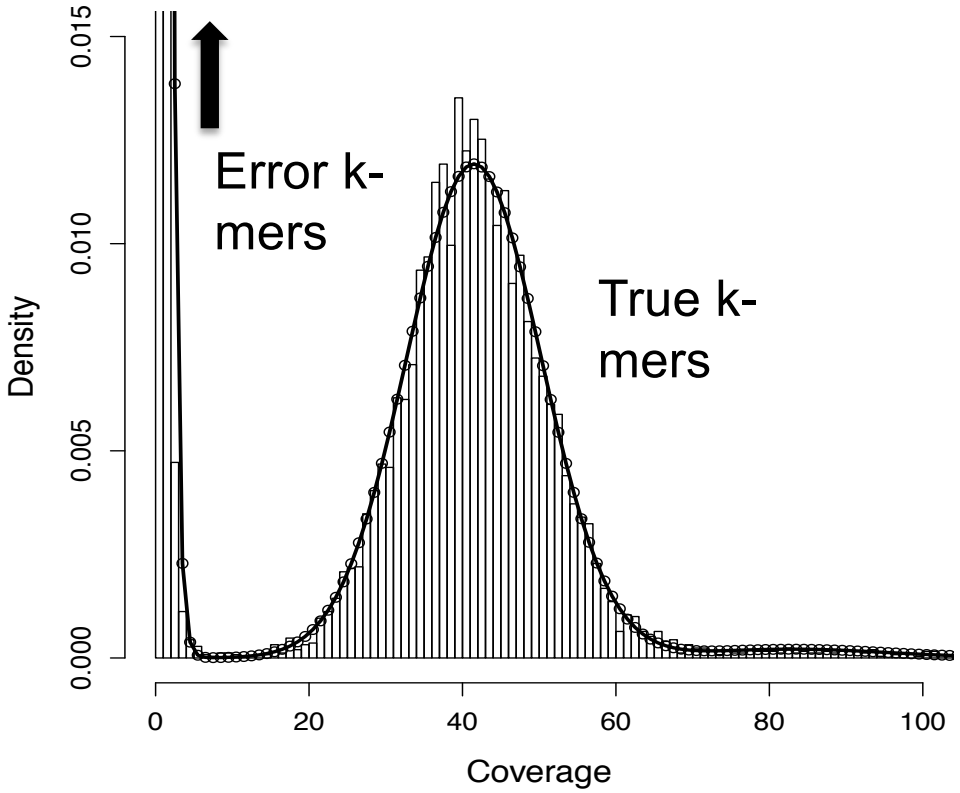
Error Correction with Quake

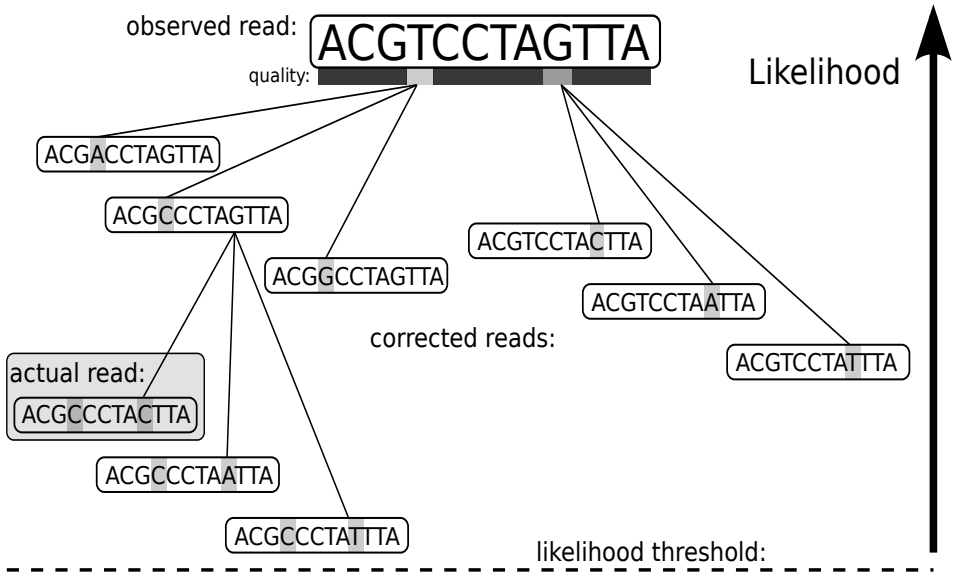
1. Count all "Q-mers" in reads

- Fit coverage distribution to mixture model of errors and regular coverage
- Automatically determines threshold for trusted k-mers

2. Correction Algorithm

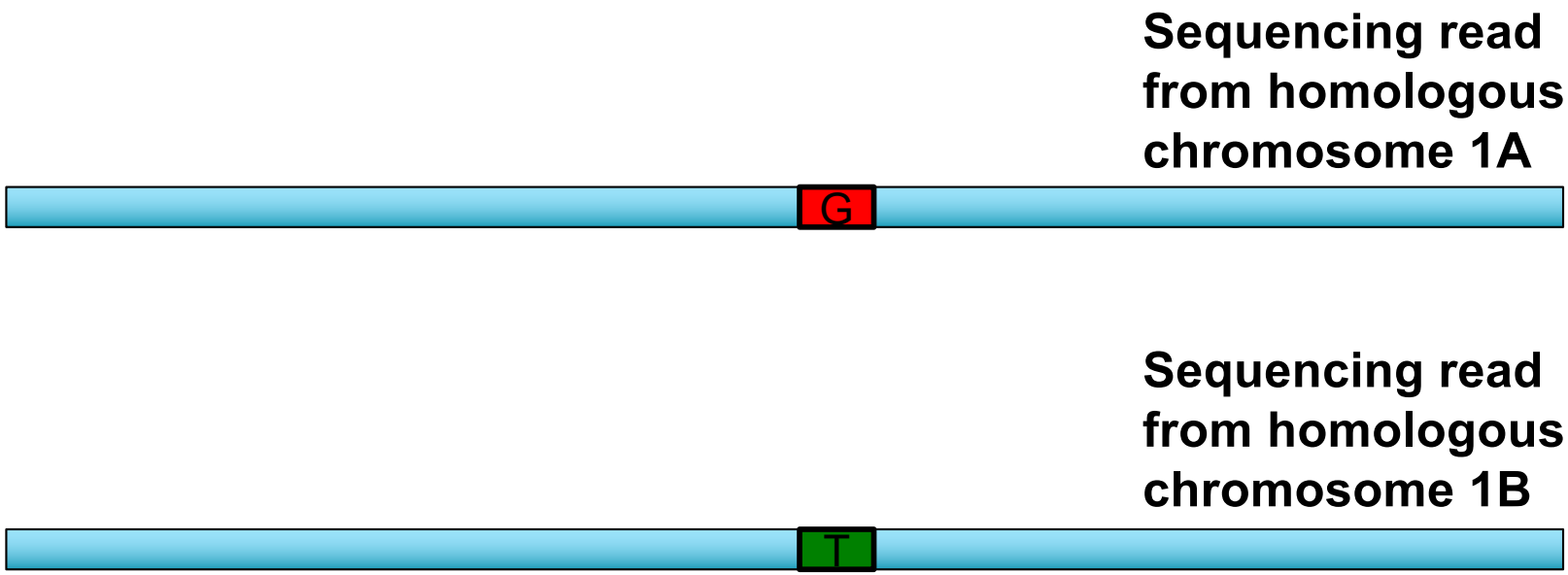
- Considers editing erroneous kmers into trusted kmers in decreasing likelihood
- Includes quality values, nucleotide/nucleotide substitution rate



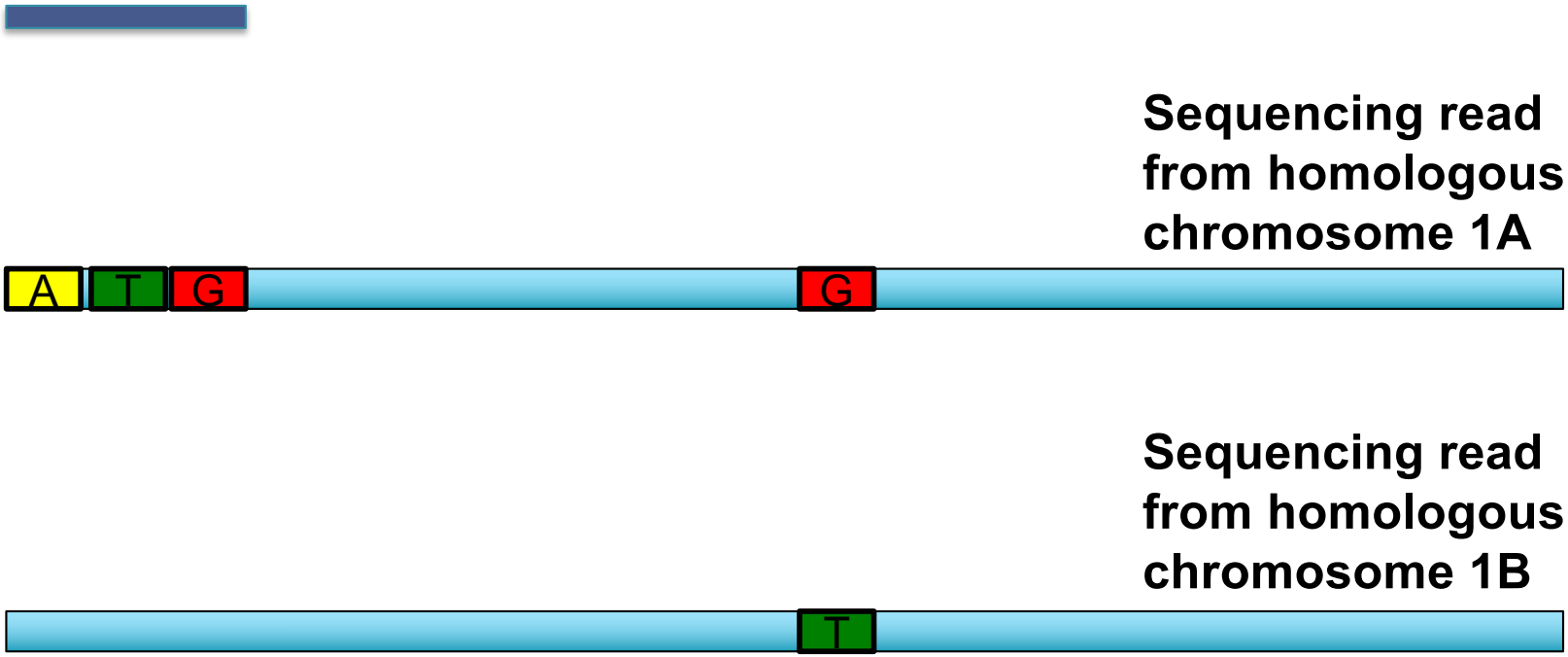


Quake: quality-aware detection and correction of sequencing reads.
Kelley, DR, Schatz, MC, Salzberg SL (2010) *Genome Biology*. 11:R116

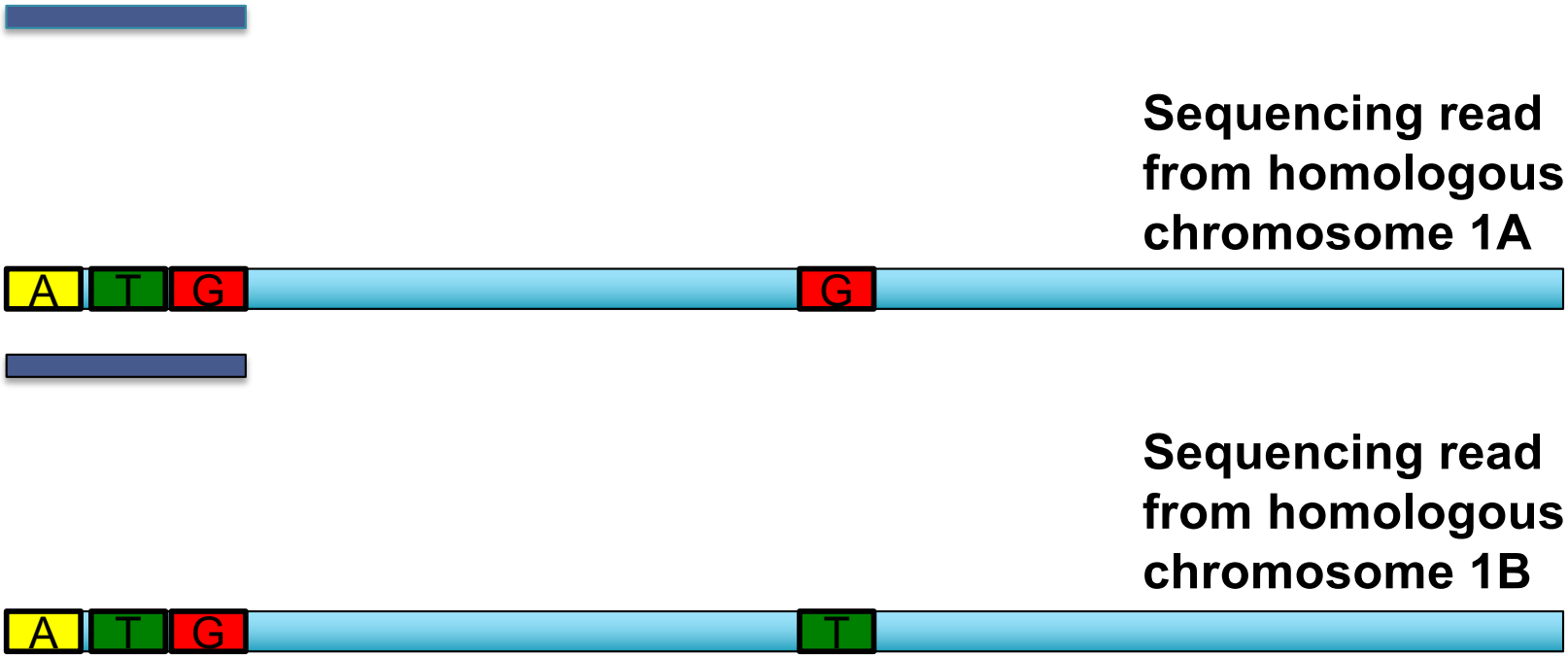
K-mer counting in heterozygous genomes



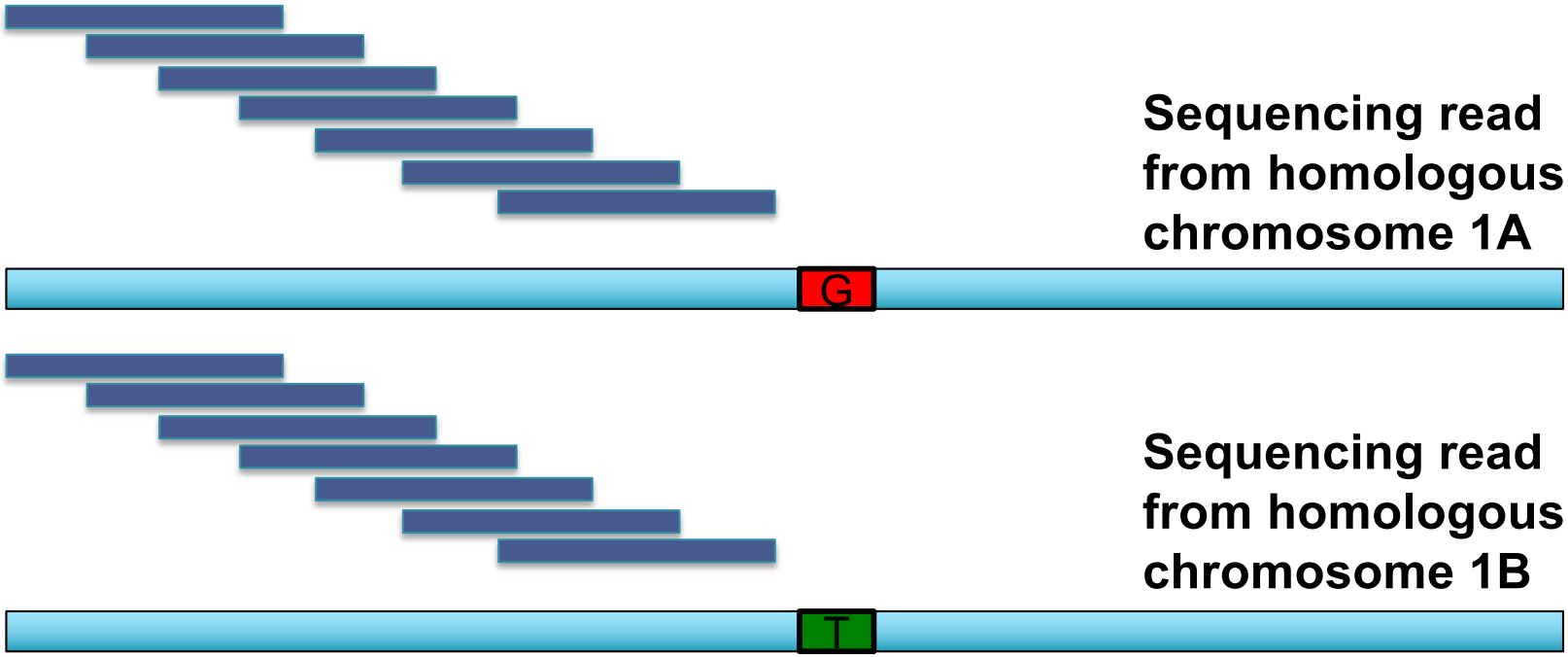
K-mer counting in heterozygous genomes



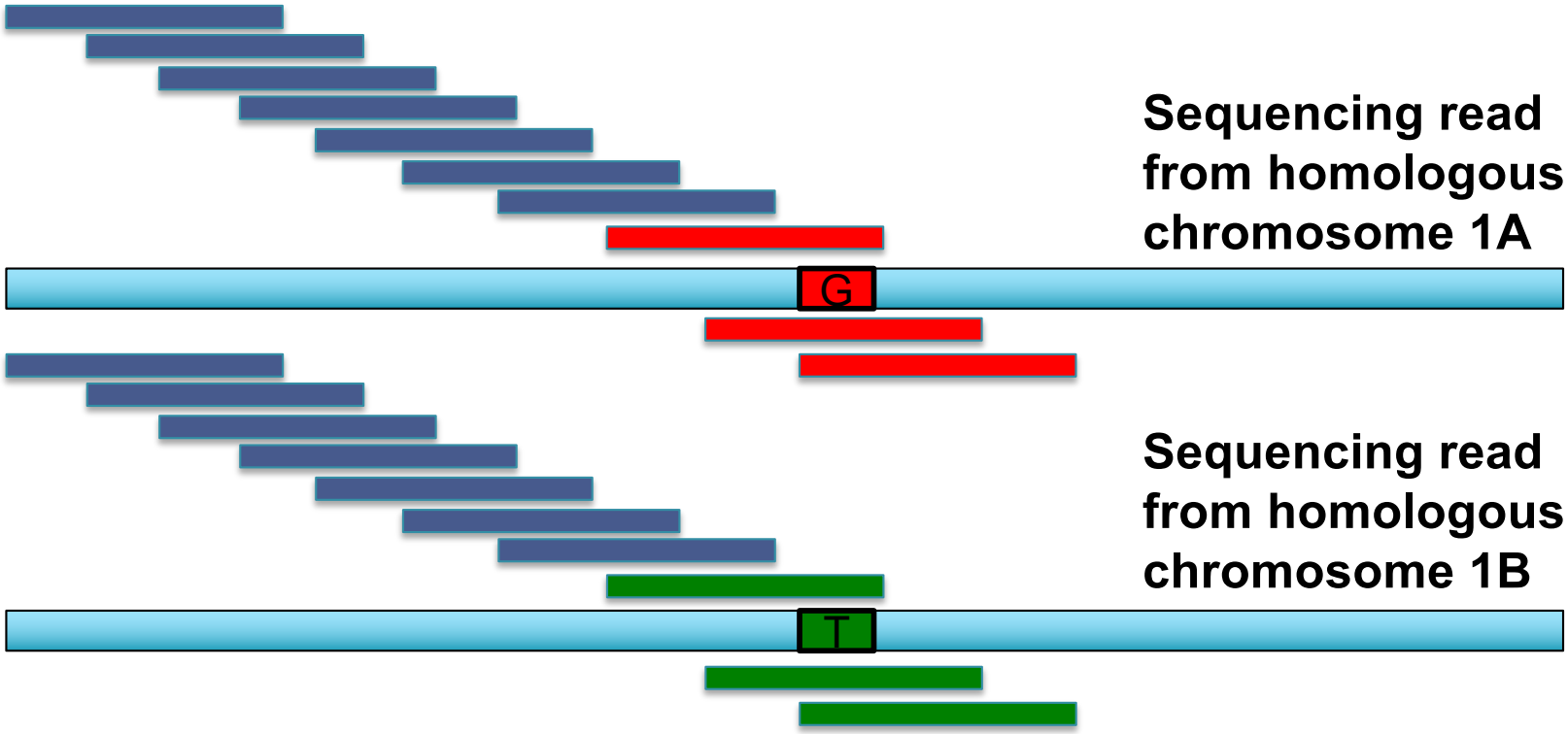
K-mer counting in heterozygous genomes



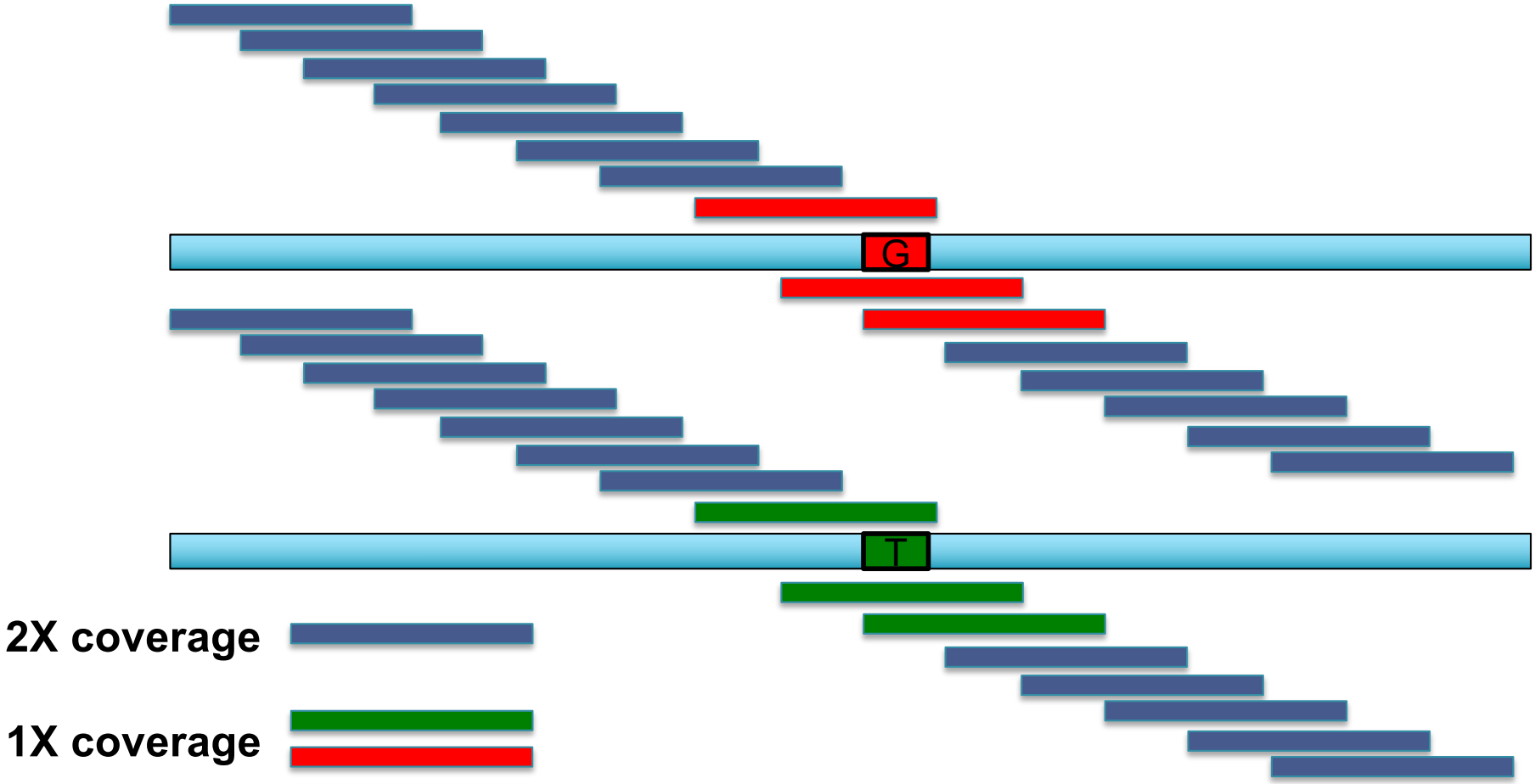
K-mer counting in heterozygous genomes



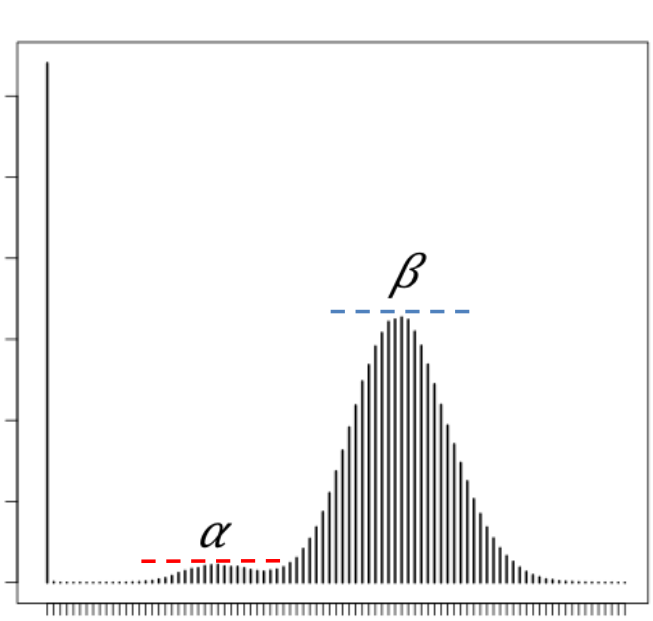
K-mer counting in heterozygous genomes



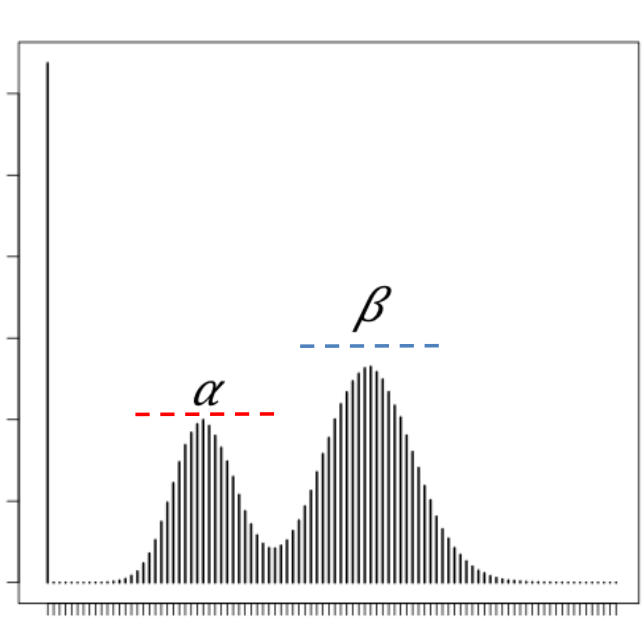
K-mer counting in heterozygous genomes



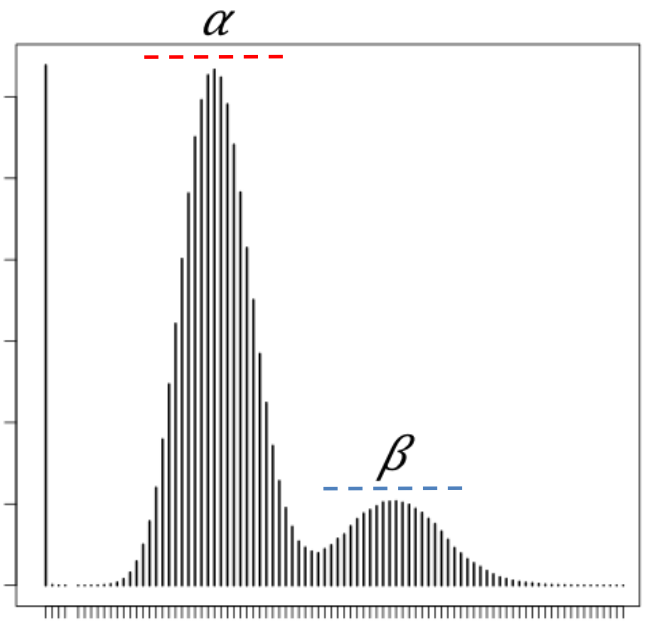
Heterozygous Kmer Profiles



0.1% heterozygosity



1% heterozygosity



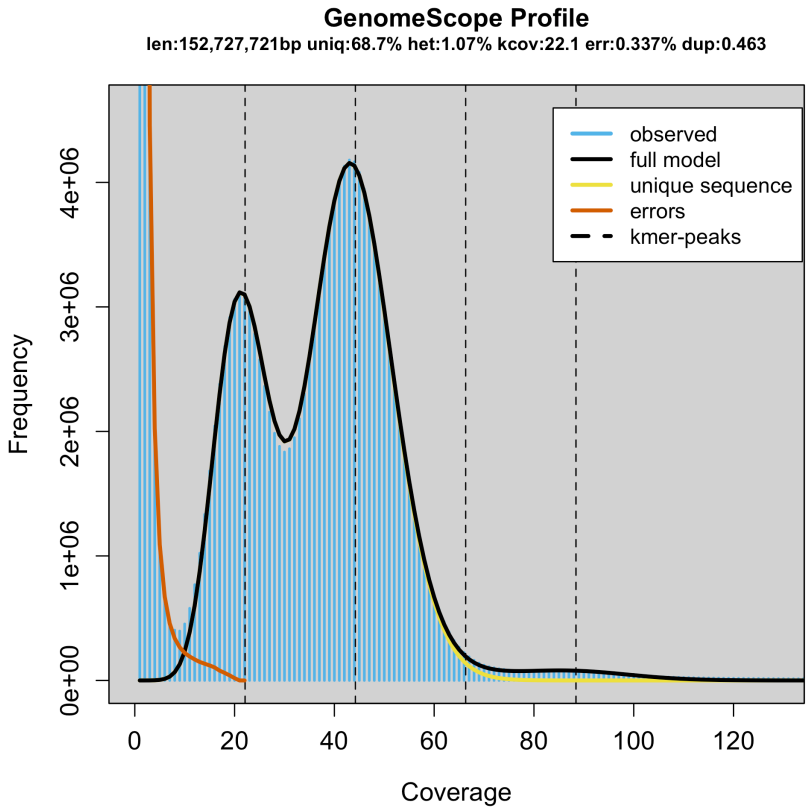
5% heterozygosity

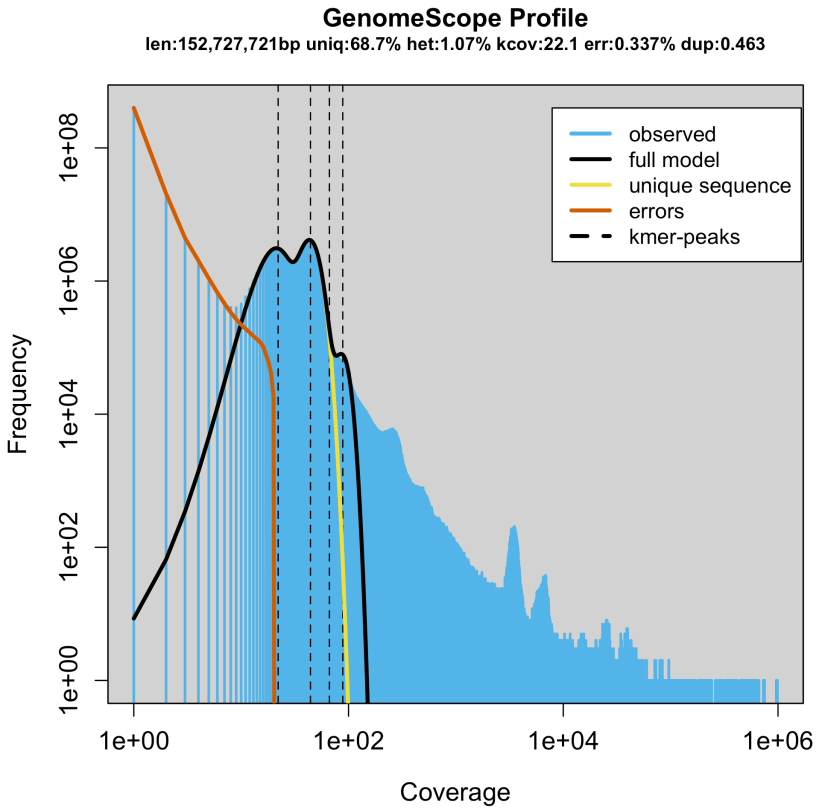
- ***Heterozygosity creates a characteristic "double-peak" in the Kmer profile***
  - Second peak at twice k-mer coverage as the first: heterozygous kmers average 50x coverage, homozygous kmers average 100x coverage

- ***Relative heights of the peaks is directly proportional to the heterozygosity rate***
  - The peaks are balanced at around 1.25% because each heterozygous SNP creates 2*k heterozygous kmers (typically k = 21)

GenomeScope: Fast genome analysis from short reads

http://genomescope.org





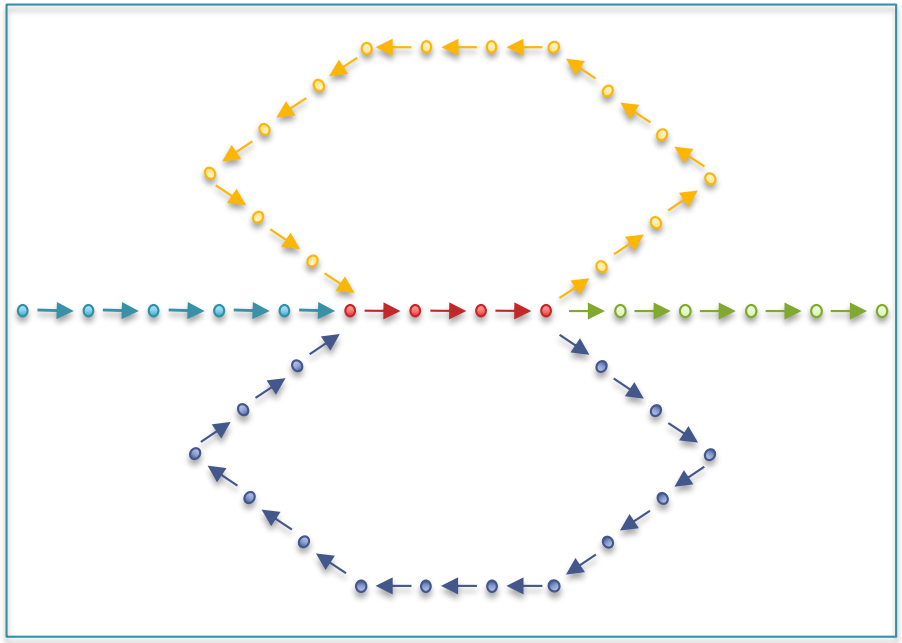
- Theoretical model agrees well with published results:
  - Rate of heterozygosity is higher than reported by other approaches but likely correct.
  - Genome size of plants inflated by organelle sequences (exclude very high freq. kmers)

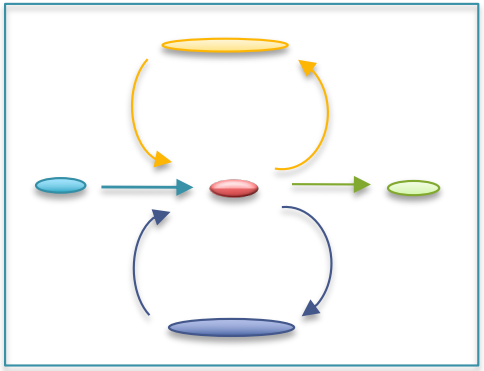
Vurture, GW*, Sedlazeck FJ*, et al. (2017) *Bioinformatics*
Ranallo-Benavidez, TR. *et al.* (2019) *bioRxiv*

Errors

Unitigging / Unipathing

- After simplification and correction, compress graph down to its non-branching initial contigs
  - Aka "unitigs", "unipaths"





Why do contigs end?

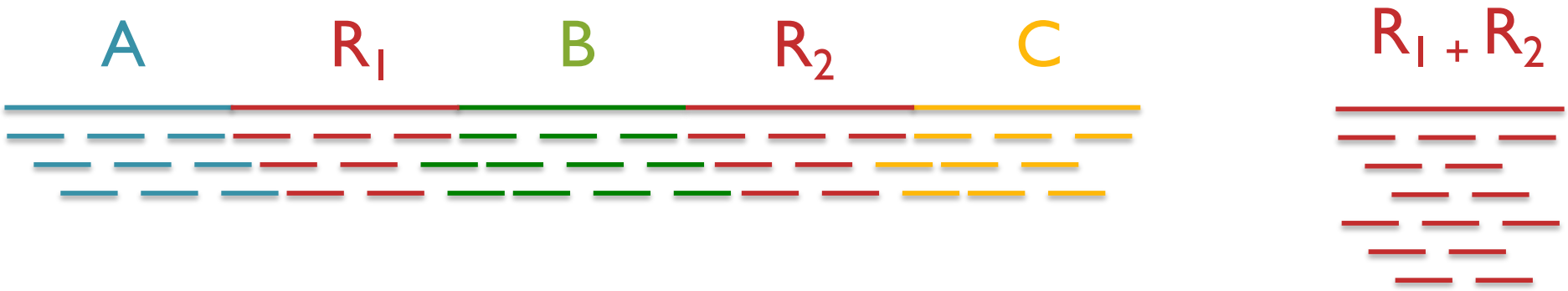
(1) End of chromosome! ☺, (2) lack of coverage, (3) errors,
(4) heterozygosity and (5) repeats

Repetitive regions

Repeat Type	Definition / Example	Prevalence
Low-complexity DNA / Microsatellites	$(b_1 b_2 \ldots b_k)^N$ where $1 \leq k \leq 6$ CACACACACACACACACACA	2%
SINEs (Short Interspersed Nuclear Elements)	*Alu* sequence (~280 bp) Mariner elements (~80 bp)	13%
LINEs (Long Interspersed Nuclear Elements)	~500 – 5,000 bp	21%
LTR (long terminal repeat) retrotransposons	Ty1-copia, Ty3-gypsy, Pao-BEL (~100 – 5,000 bp)	8%
Other DNA transposons		3%
Gene families & segmental duplications		4%

- Over 50% of mammalian genomes are repetitive
  - Large plant genomes tend to be even worse
  - Wheat: 16 Gbp; Pine: 24 Gbp

Repeats and Coverage Statistics



- If *n* reads are a uniform random sample of the genome of length *G*, we expect $k=n\Delta/G$ reads to start in a region of length Δ.
  - If we see many more reads than k (if the arrival rate is > A) , it is likely to be a collapsed repeat

$$\Pr(X-copy)=\binom{n}{k}\left(\frac{X\Delta}{G}\right)^{k}\left(\frac{G-X\Delta}{G}\right)^{n-k}$$

$$A(\Delta,k)=\ln\left(\frac{\Pr(1-copy)}{\Pr(2-copy)}\right)=\ln\left(\frac{\frac{(\Delta n/G)^{k}}{k!}e^{\frac{-\Delta n}{G}}}{\frac{(2\Delta n/G)^{k}}{k!}e^{\frac{-2\Delta n}{G}}}\right)=\frac{n\Delta}{G}-k\ln 2$$

The fragment assembly string graph
Myers, EW (2005) Bioinformatics. 21(suppl 2): ii79-85.

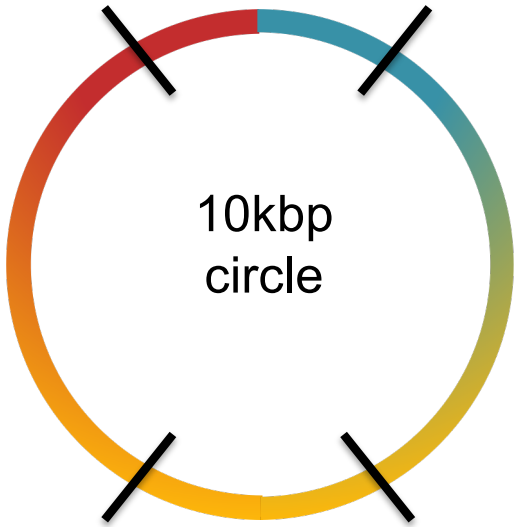
Paired-end and Mate-pairs

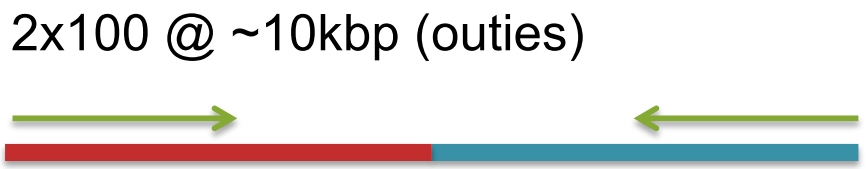
Paired-end sequencing

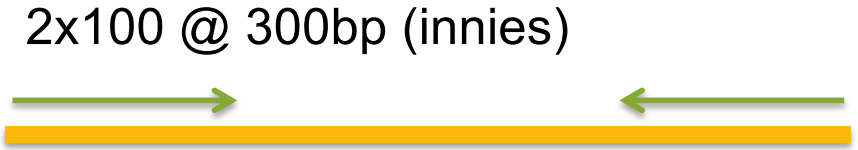
- Read one end of the molecule, flip, and read the other end
- Generate pair of reads separated by up to 500bp with inward orientation

Mate-pair sequencing

- Circularize long molecules (1-10kbp), shear into fragments, & sequence
- Mate failures create short paired-end reads

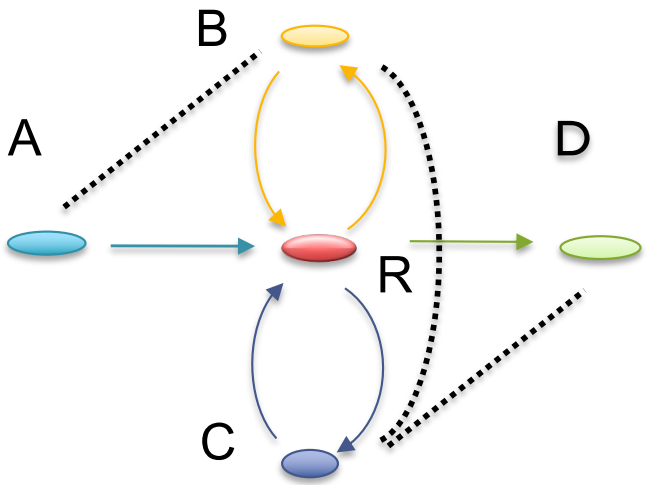




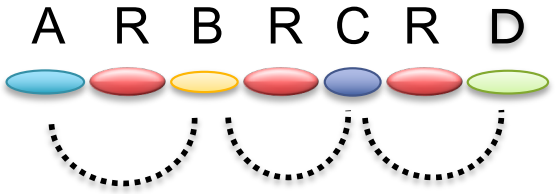


Scaffolding

- Initial contigs (*aka* unipaths, unitigs) terminate at
  - *Coverage gaps*: especially extreme GC
  - *Conflicts*: errors, repeat boundaries



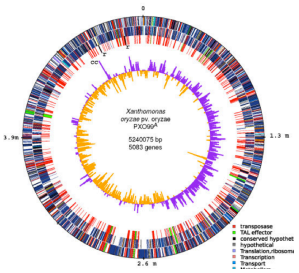
- Use mate-pairs to resolve correct order through assembly graph
  - Place sequence to satisfy the mate constraints
  - Mates through repeat nodes are tangled



- Final scaffold may have internal gaps called sequencing gaps
  - We know the order, orientation, and spacing, but just not the bases. Fill with Ns instead

Why do scaffolds end?

Assembly Summary



Assembly quality depends on

1. ***Coverage***: low coverage is mathematically hopeless
2. ***Repeat composition***: high repeat content is challenging
3. ***Read length***: longer reads help resolve repeats
4. ***Error rate***: errors reduce coverage, obscure true overlaps

- Assembly is a hierarchical, starting from individual reads, build high confidence contigs/unitigs, incorporate the mates to build scaffolds
  - Extensive error correction is the key to getting the best assembly possible from a given data set

- Watch out for collapsed repeats & other misassemblies
  - Globally/Locally reassemble data from scratch with better parameters & stitch the 2 assemblies together

Next Steps

1. Reflect on the magic and power of DNA ☺
2. Check out the course webpage
3. Register on Piazza
4. Work on Assignment 1
    1. Set up Linux, set up Virtual Machine
    2. Set up Dropbox for yourself!
    3. Get comfortable on the command line