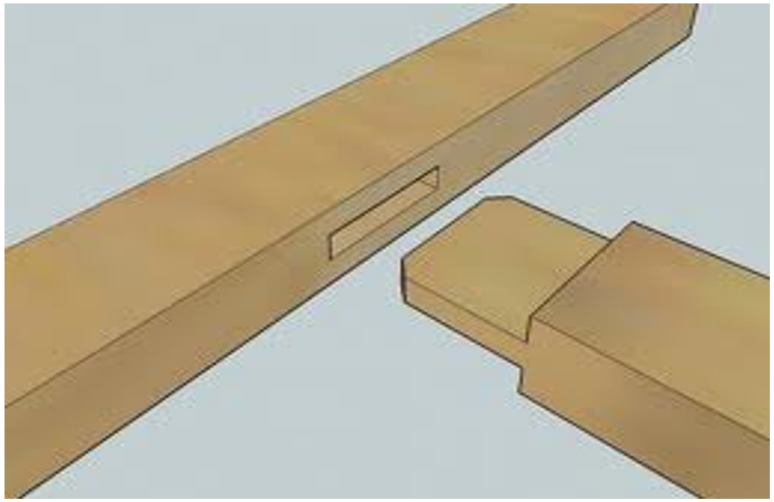
ASSEMBLY ALGORITHMS

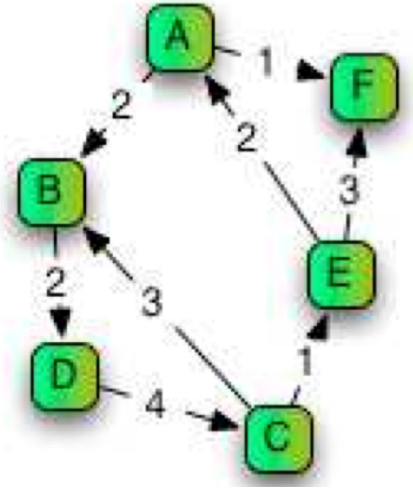


Sources :

- GTN : Deeper look into Genome Assembly algorithms
- The Fundamentals of Genome Assembly : Jared Simpson . High--Throughput Biology: From Sequence to Networks
- Genome properties and De Novo Genome Assembly - Introduction Henrik Lantz - NBIS/SciLife/Uppsala University
- Genome Assembly: Chris Fields Mayo- Illinois Computational Genomics Workshop, June 19, 2017
- Long Reads : *"Chasing perfection".* Claude THERMES. 10eme Ecole de bioinformatique EBAII - 23/11/2021

What is a graph ?

- Not an Excel chart!
- Nodes/Vertices
  - A,B,E,G,H,K,M
- Edges/Arcs
  - (lines between nodes)
- Directed graph
  - Arrow head on edge
- Weighted graph
  - Numerals on edges



Overlap - Layout – Consensus

- Overlap
  - All against all pair-wise comparison
  - Build graph: nodes=reads, edges=overlaps
- Layout
  - Analyse/simplify/clean the overlap graph
  - Determine Hamiltonian path (NP-hard) : a graph having at least one cycle passing through all vertices once and only once.
- Consensus
  - Align reads along assembly path
  - Call bases using weighted voting

OLC : Pairwise Overlap

- All against all pair-wise comparison
  - 1⁄2 N(N-1) alignments to perform [N=no. reads]
- In practice, use smarter heuristics
  - Index all k-mers from all reads
  - Only check pairs that share enough k-mers
  - Similar approach to BLAST algorithm
- Both approaches parallelizable
  - Each comparison is independent

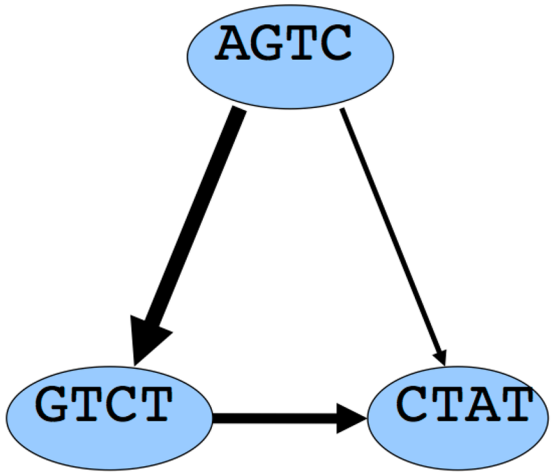
OLC: Overlap Example

- True sequence (7bp) : AGTCTAT
- Reads (3 x 4bp) : AGTC, GTCT, CTAT
- Pairs to align (3)
  AGTC+GTCT , AGTC+CTAT , GTCT+CTAT
- Best overlaps

```
AGTC-        AGTC---      GTCT—
-GTCT        ---CTAT      --CTAT
(good)       (poor)        (ok)
```

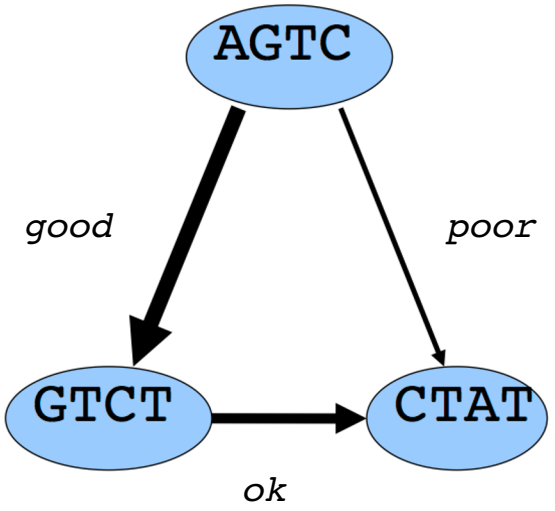
OLC: Overlap Graph

- Nodes are the 3 reads sequences
- Edges are the overlap alignment with orientation



OLC: Overlap Graph

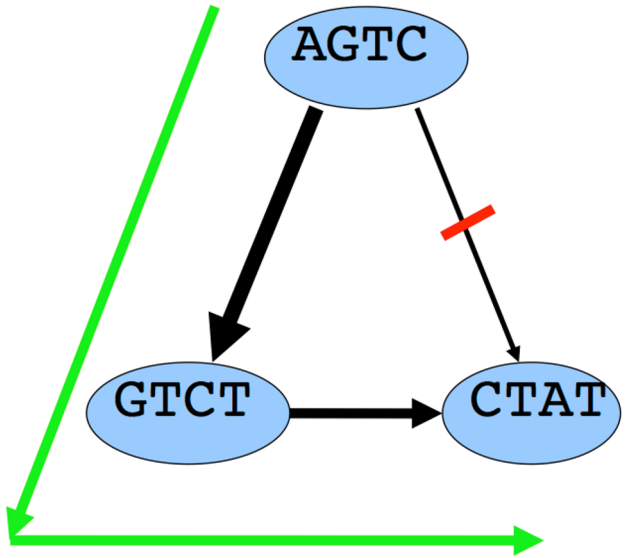
- Nodes are the 3 reads sequences
- Edges are the overlap alignment with orientation
- Edge thickness represents score of overlap




```
AGTC-      AGTC---      GTCT—
-GTCT      ---CTAT      --CTAT
(good)     (poor)        (ok)
```

OLC: Layout – Consensus

- Optimal path shown in green
- Un-traversed weak overlap in red
- Consensus is read by outputting the overlapped nodes along the path
- `aGTCTCTat`



OLC : Softwares

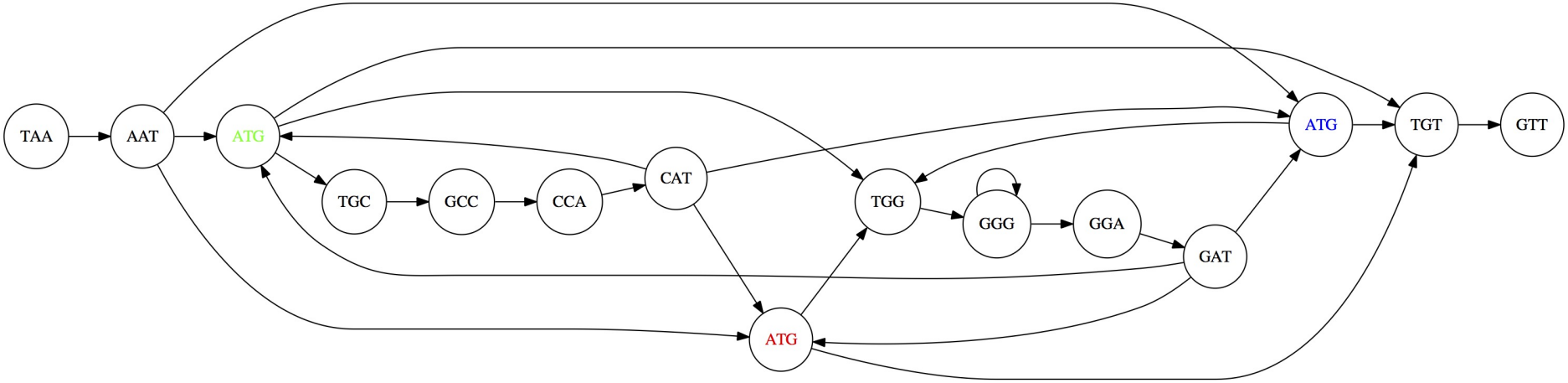
- Phrap, PCAP, CAP3
  - Smaller scale assemblers
- Celera Assembler
  - Sanger-era assembler for large genomes
- Arachne, Edena, CABOG, Mira 4
  - Modern Sanger/hybrid assemblers

- **Newbler (gsAssembler)**
  - Used for 454 NGS “long” reads
  - Can be used for IonTorrent flowgrams too

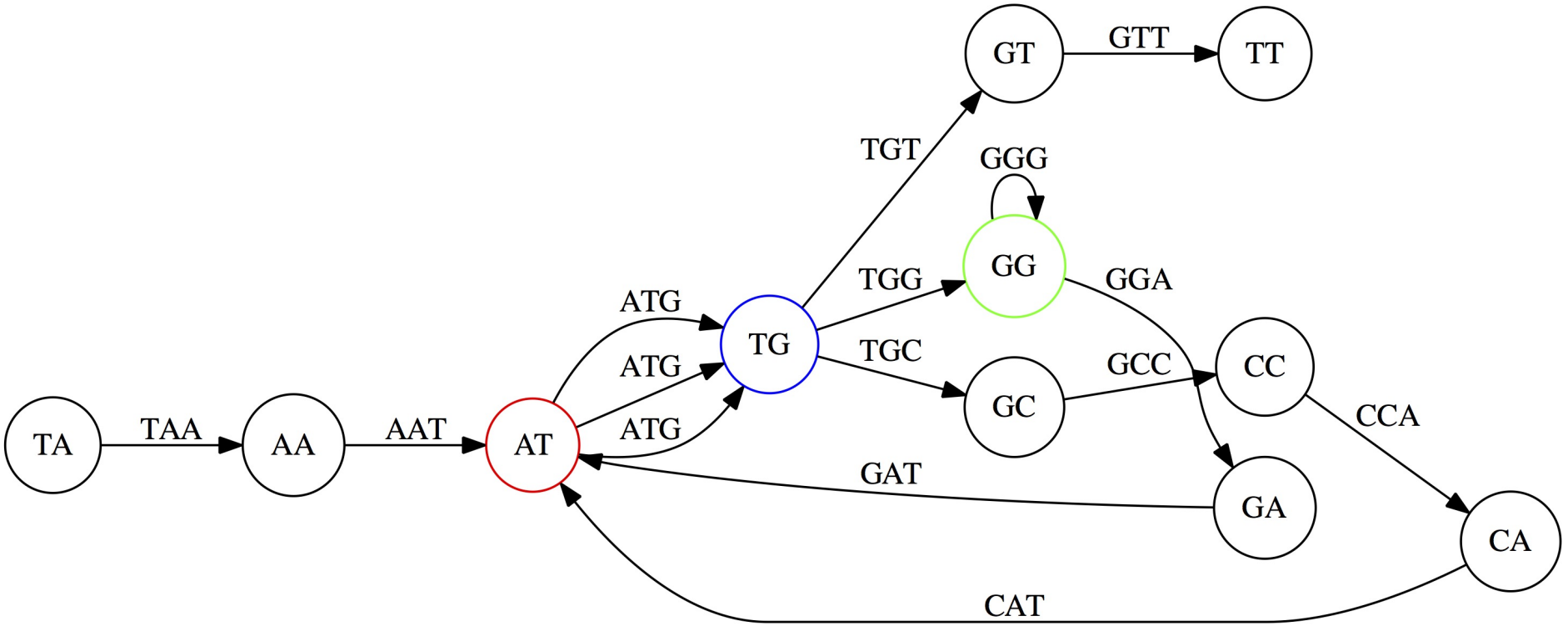
Eulerian approach

- Break all reads (length L) into (L-k+1) k-mers
  - L=36, k=31 gives 6 k-mers per read
- Construct a *de Bruijn* graph (DBG)
  - Nodes = one for each unique k-mer
  - Edges = k-1 exact overlap between two nodes
- Graph simplification
  - Merge chains, remove bubbles and tips
- Find a Eulerian path through the graph : where we pass through all the edges once and only once
  - Linear time algorithm, unlike Hamiltonian

Overlap graph



de Bruijn Graph - same data



de Bruijn graph assembly process

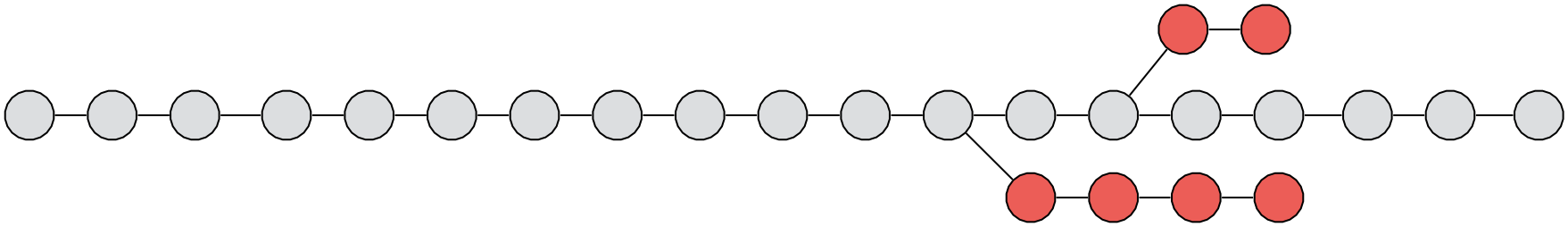
1. Select a value for k
2. "Hash" the reads (make the kmers)
3. Count the kmers
4. Make the de Bruijn graph
5. Perform graph simplification steps
6. Read off contigs from simplified graph

DBG: graph simplification

- Remove tips or spurs
  - Dead ends in graph due to errors at read end
- Collapse bubbles
  - Errors in middle of reads
  - But could be true SNPs or diploidity
- Remove low coverage paths
  - Possible contamination
- Makes final Eulerian path easier
  - And hopefully more accurate contigs

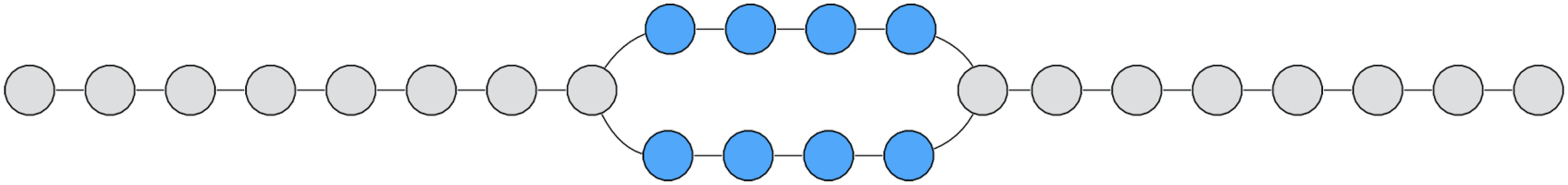
Graph Artefacts - Tips

$Read_1$	GTATCGATCGACTAGCTACGACTAGCTACGATCGACTACGATCA
$Read_2$	TCGATCGACTAGCTACGACTAGCTACGATCGACTACGAACAGC
$Read_3$	GATCGACTAGCTACGACTAGCTACGATCGACTACGATCGGCATC

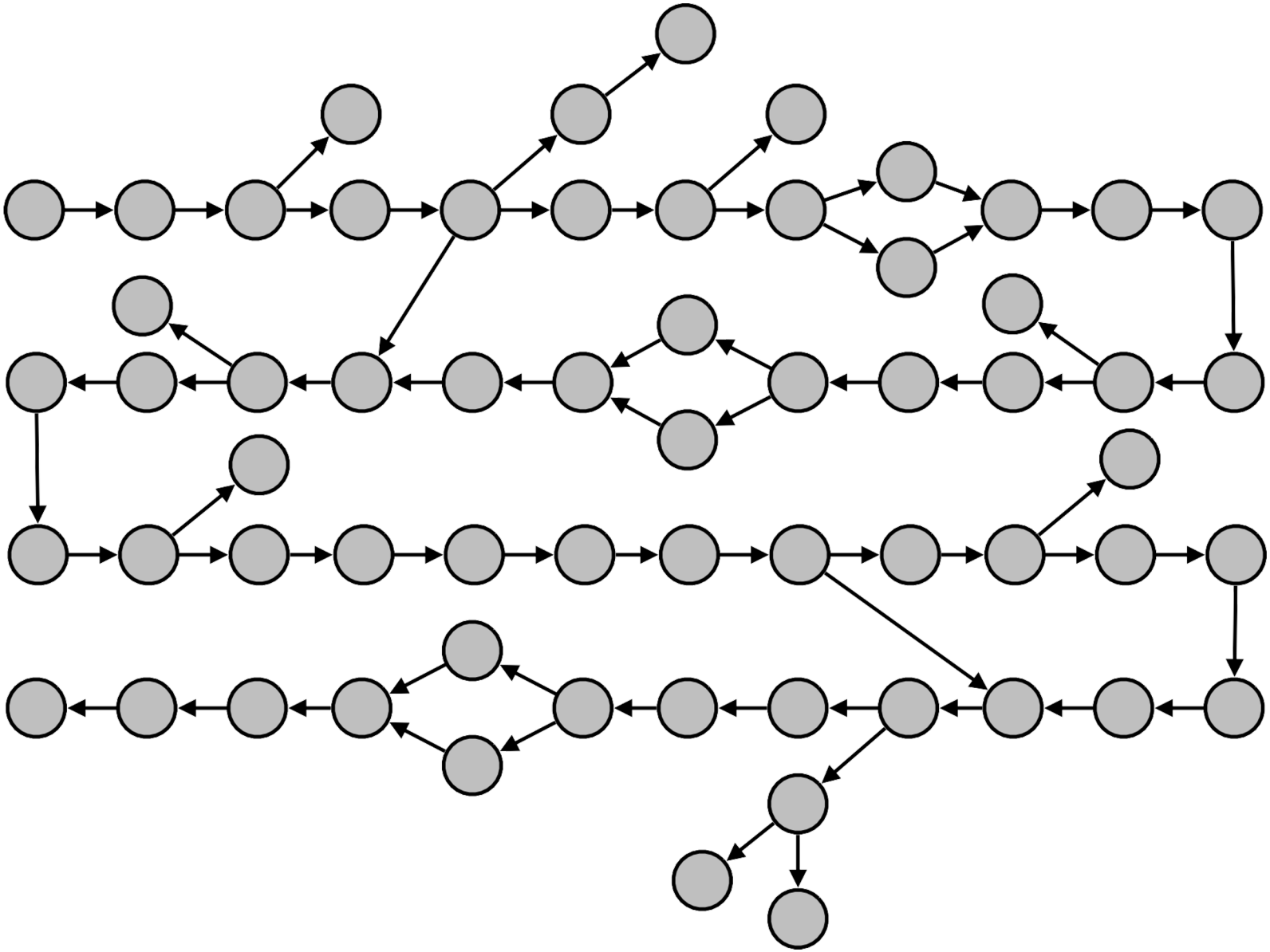


Graph Artefacts - Bubbles

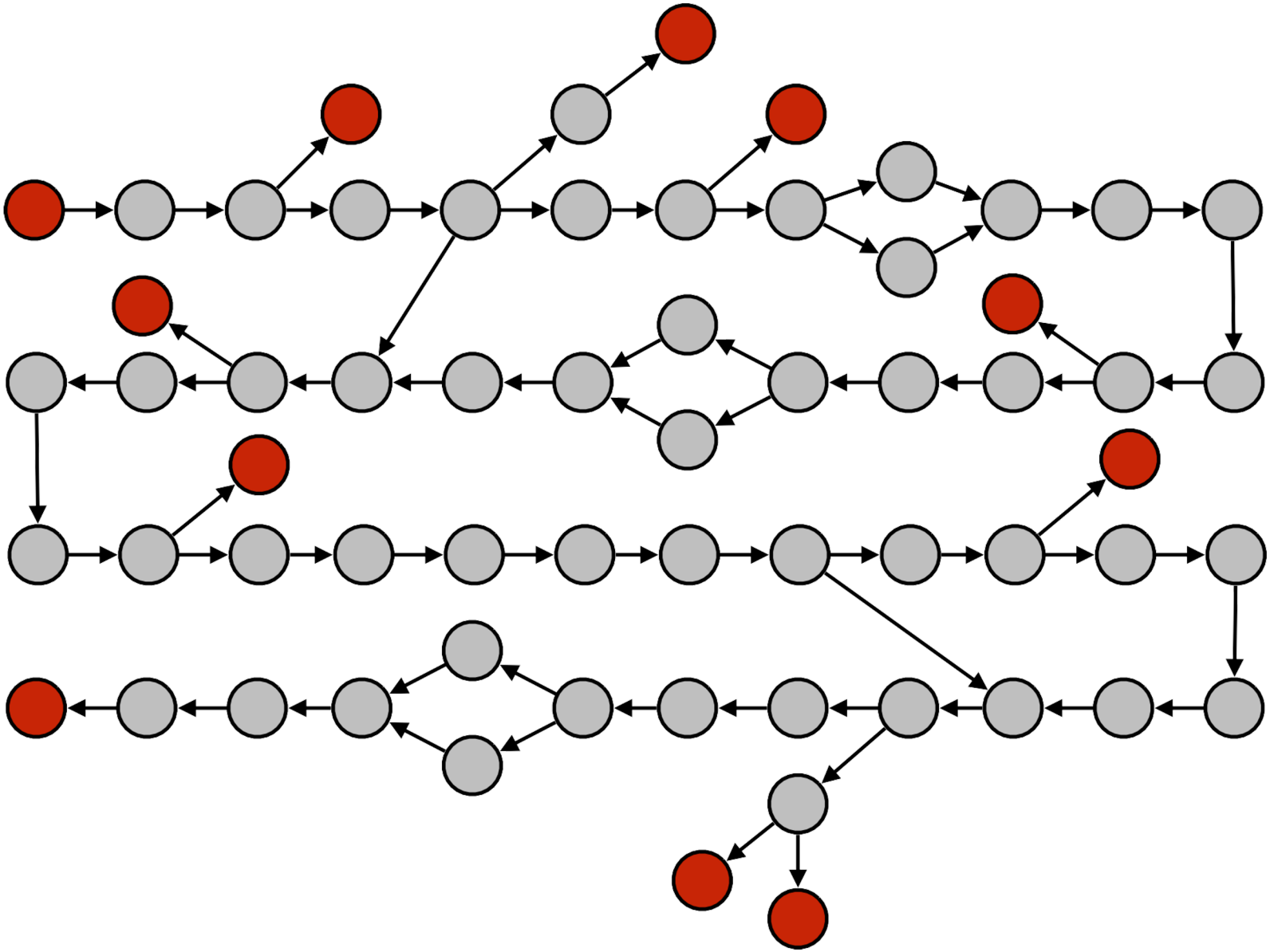
$Allele_1$	GACTAGCTATATCGATCGATCGATCGATCTCTAGACTACGACTGAAATC
$Allele_2$	GACTAGCTATATCGATCGATCGATGGATCTCTAGACTACGACTGAAATC



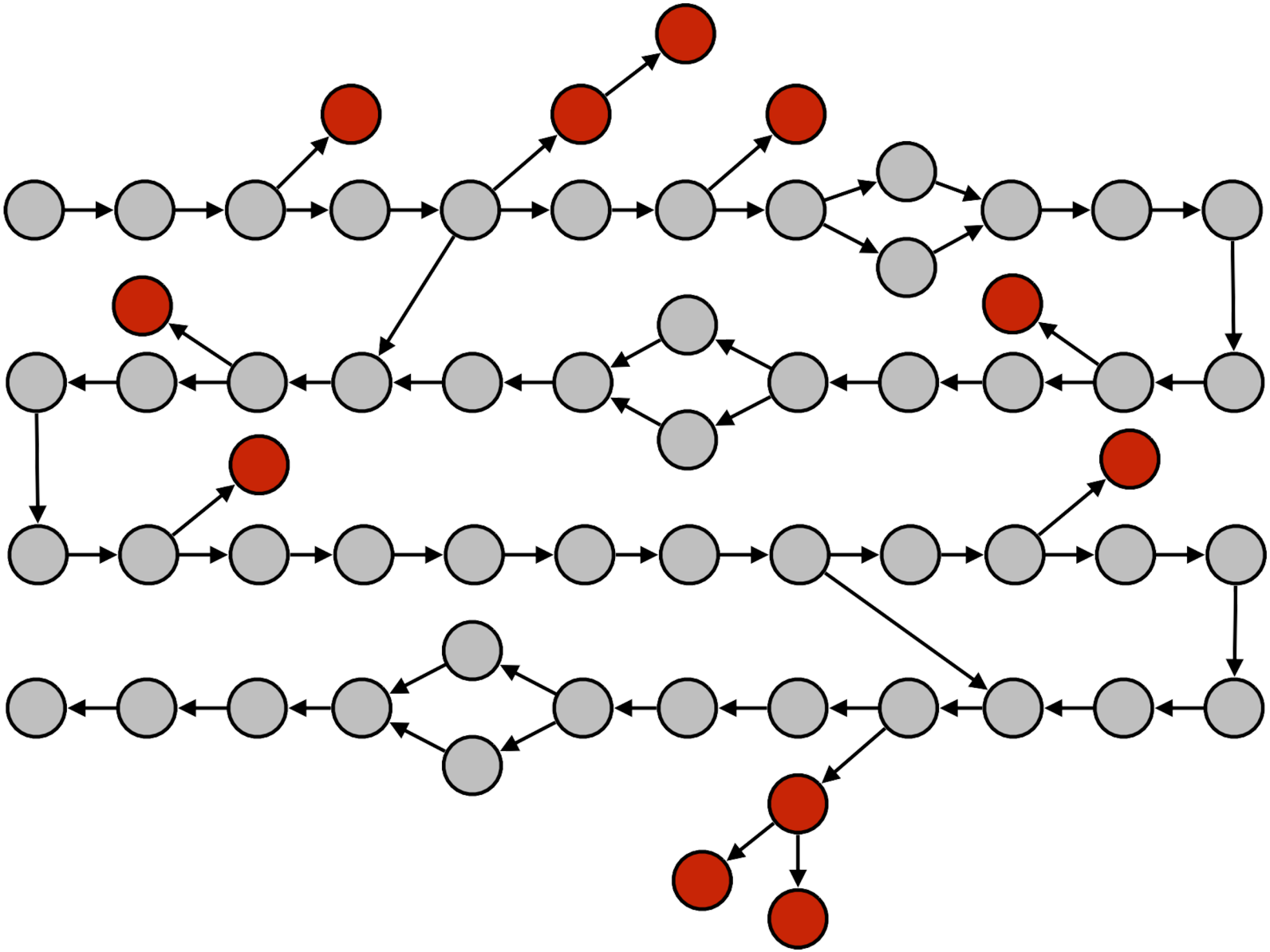
Graph Cleaning



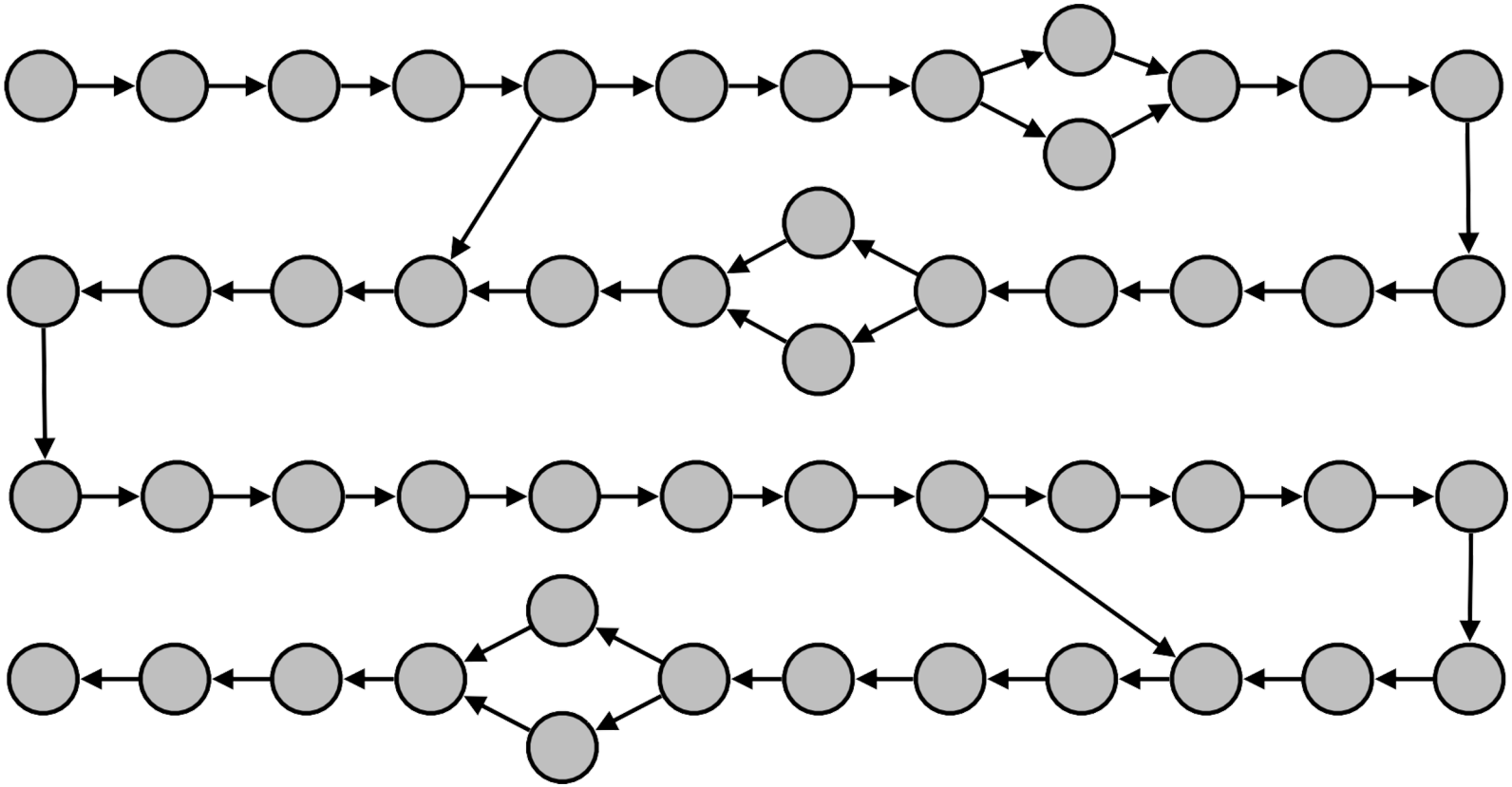
Tips removal



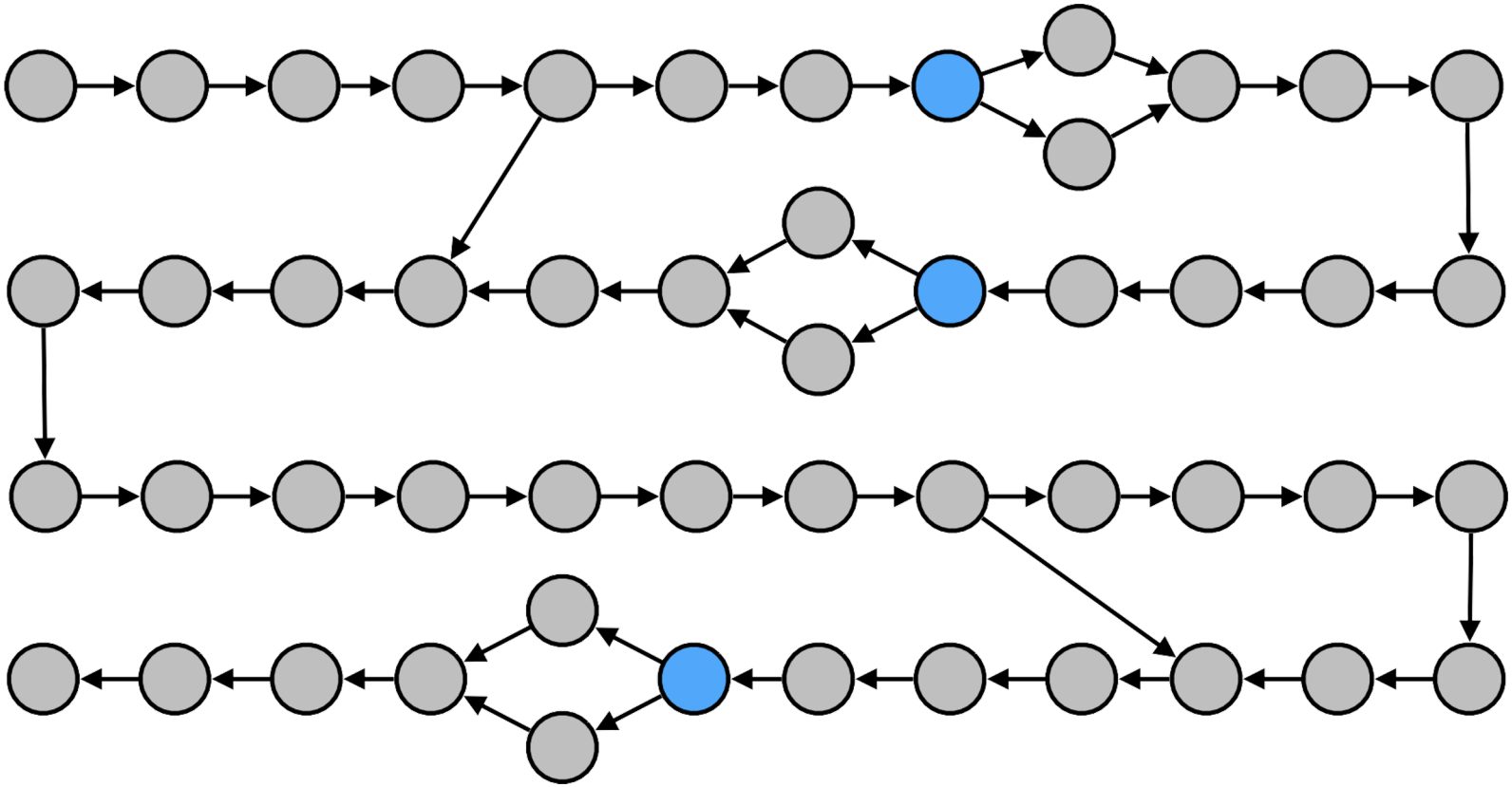
Tips removal



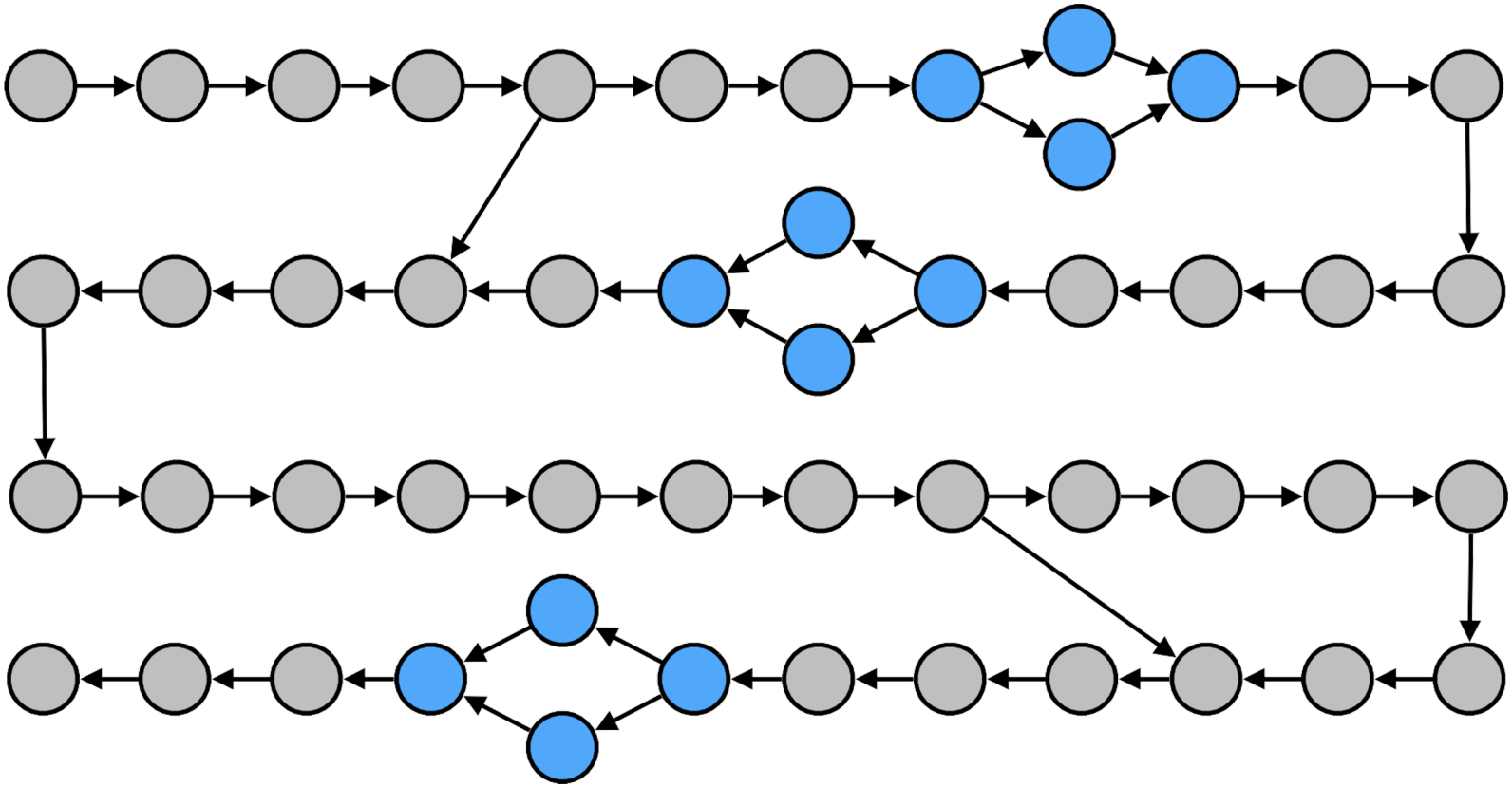
Tips removal



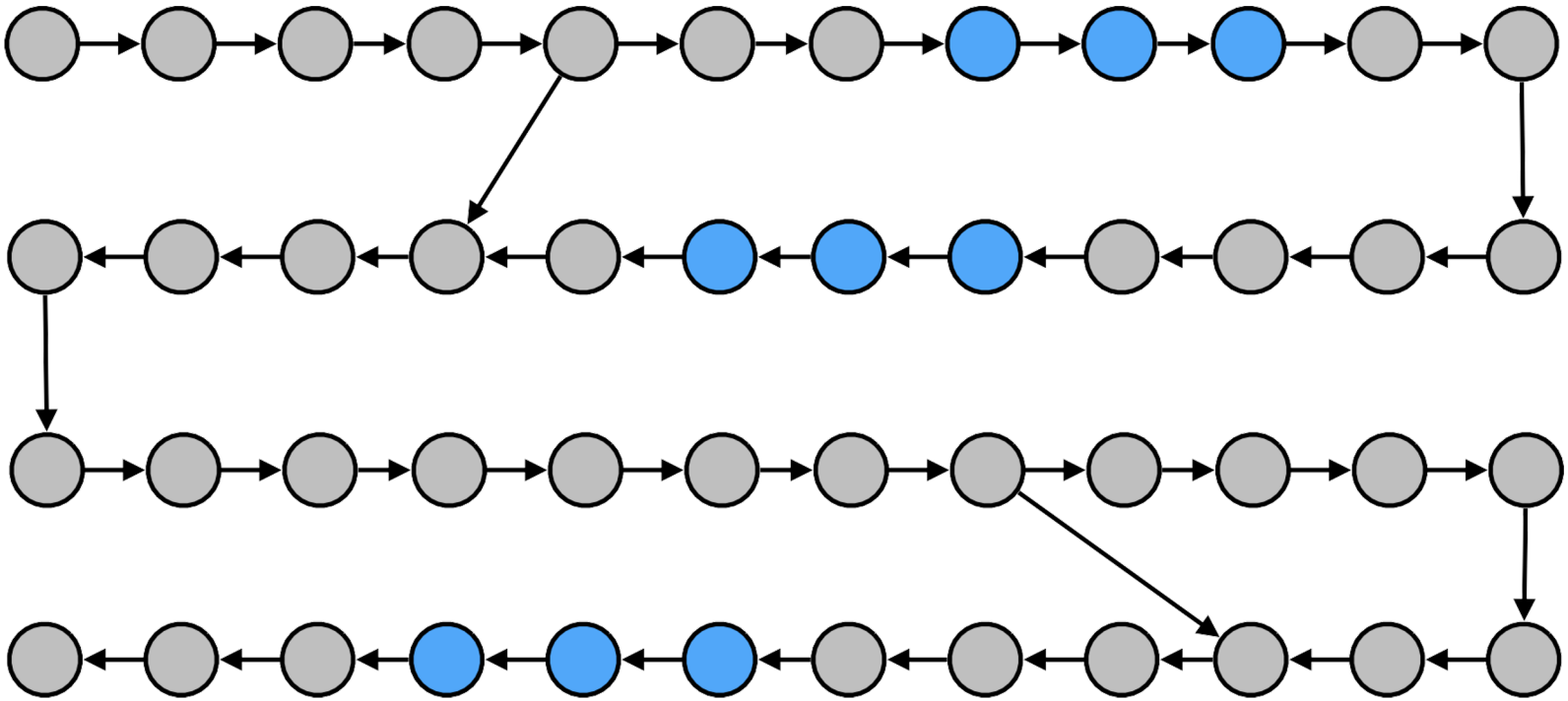
Bubble Removal



Bubble Removal



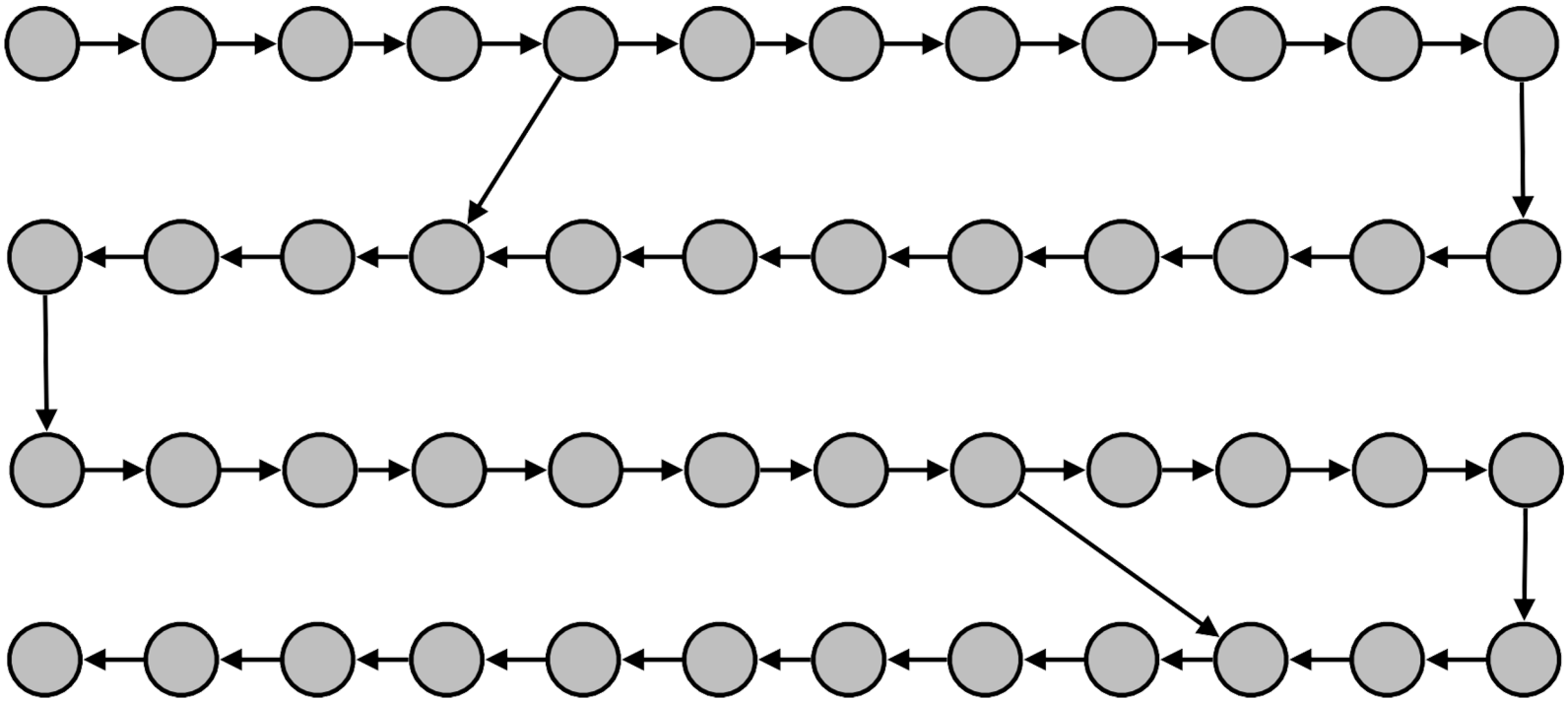
Bubble Removal



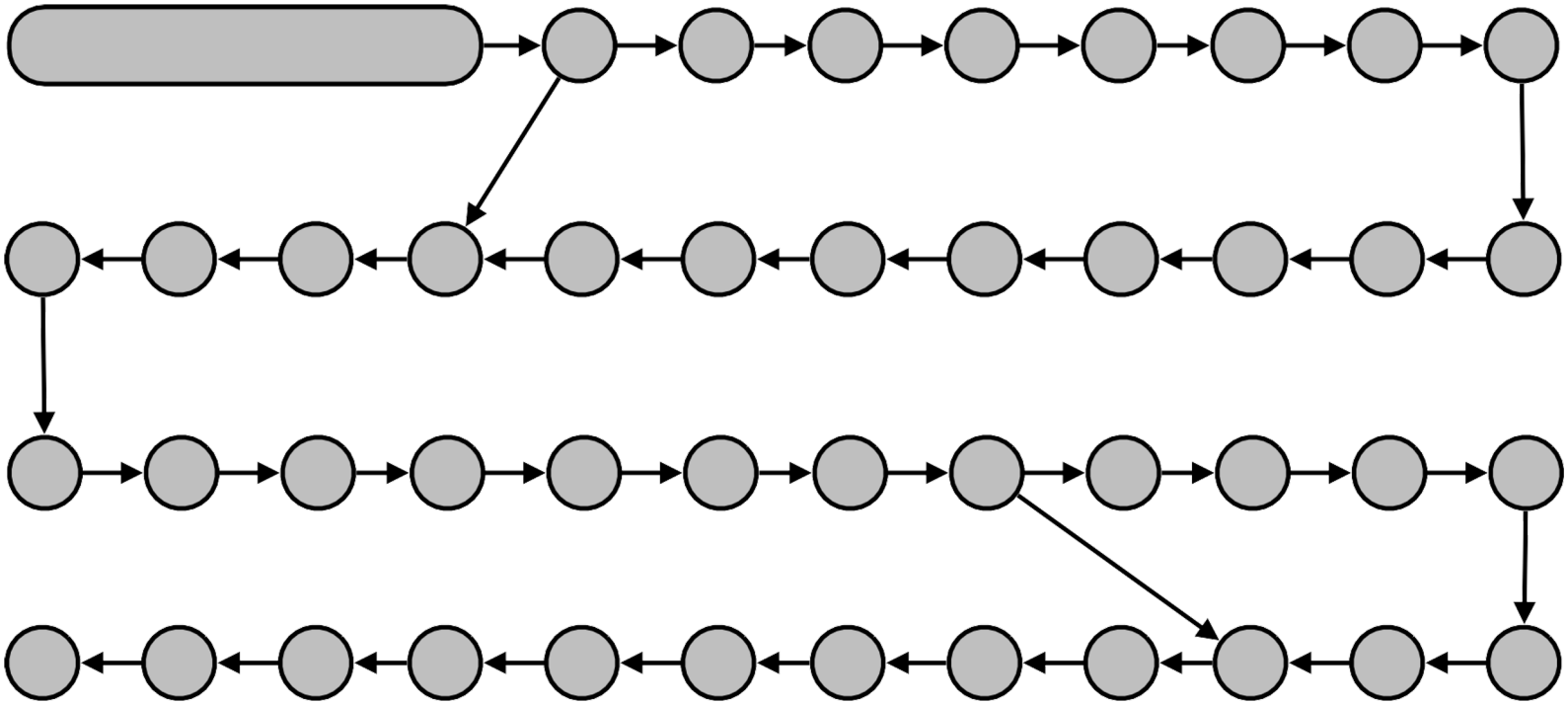
Make contigs

- Find an unbalanced node in the graph
- Follow the chain of nodes and "read off" the bases to produce the contigs
- Where there is an ambiguous divergence/convergence, stop the current contig and start a new one.
- Re-trace the reads through the contigs to help with repeat resolution

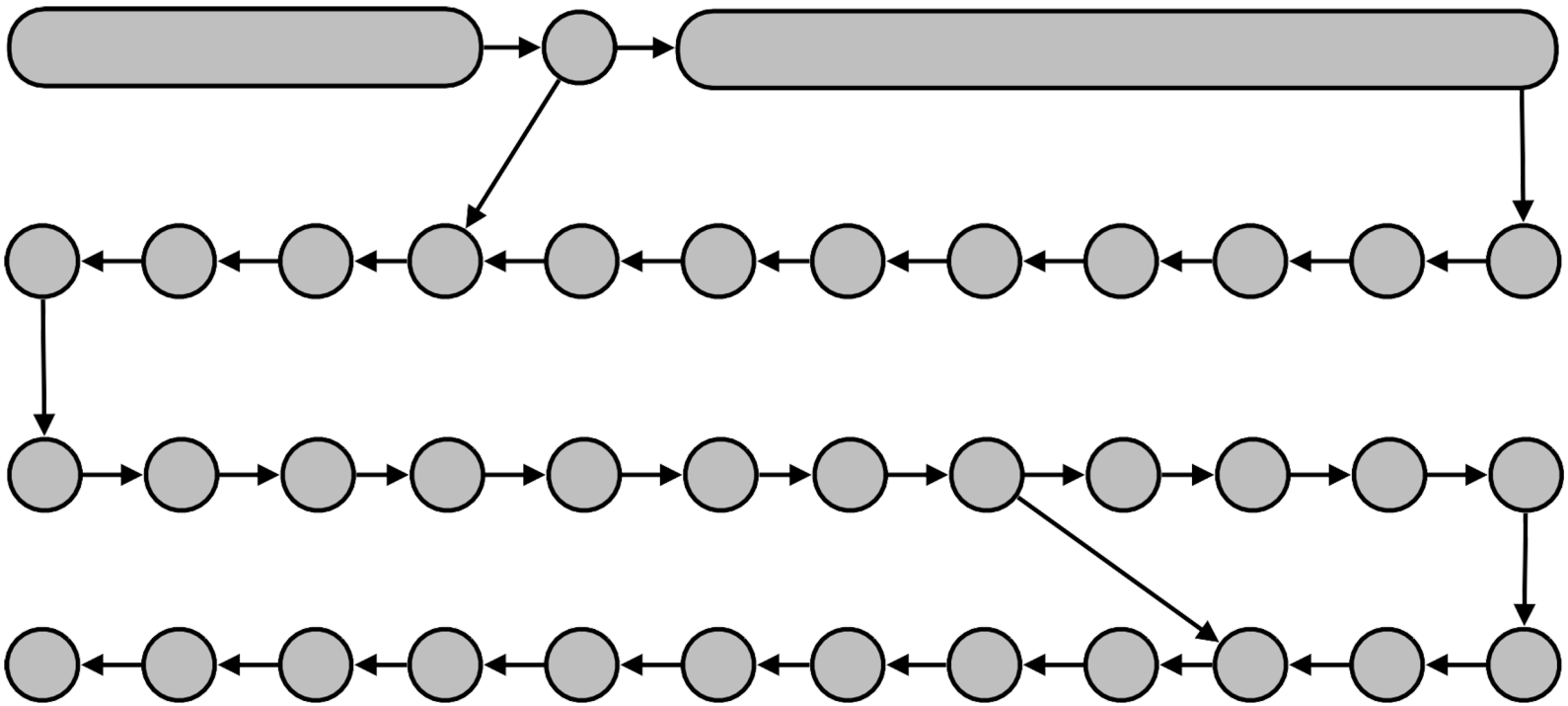
Contig Assembly



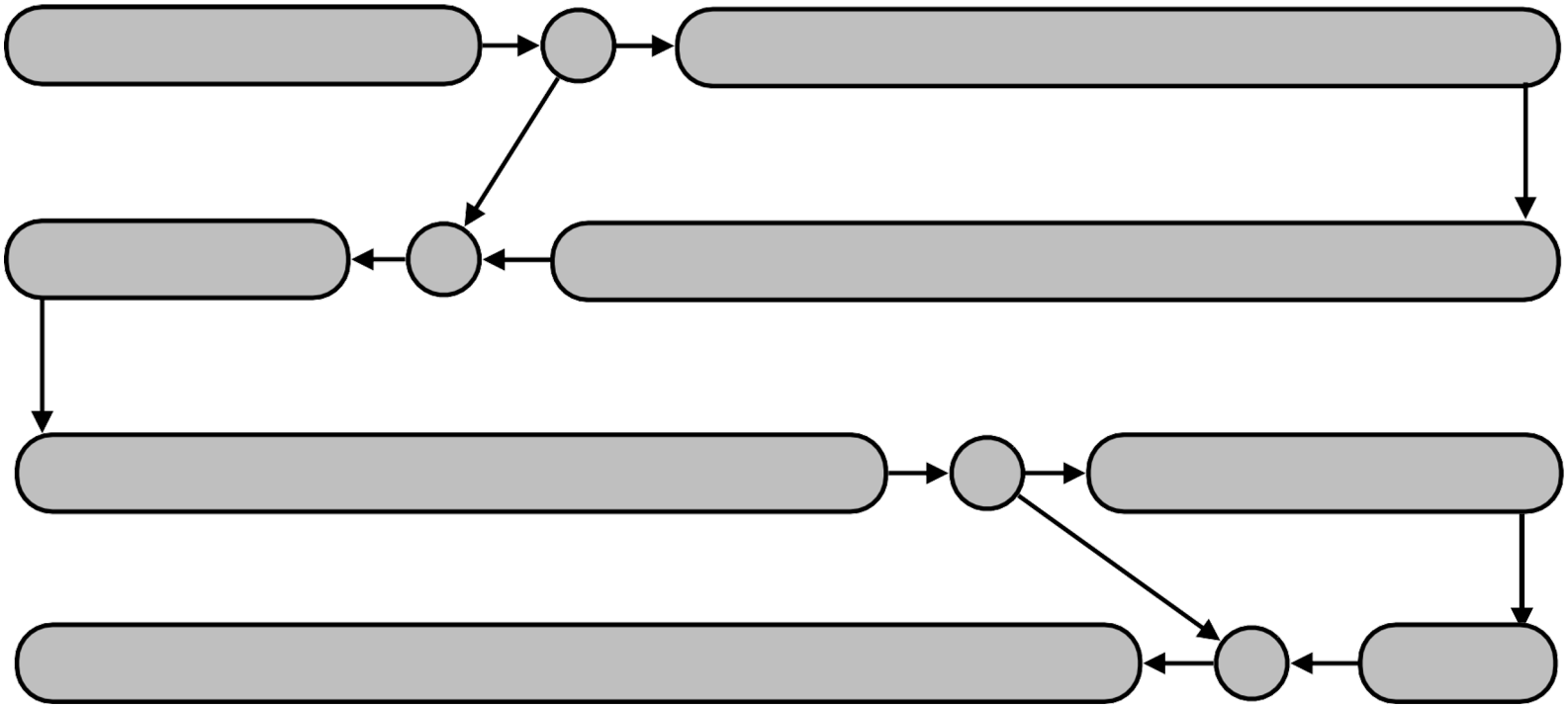
Contig Assembly



Contig Assembly

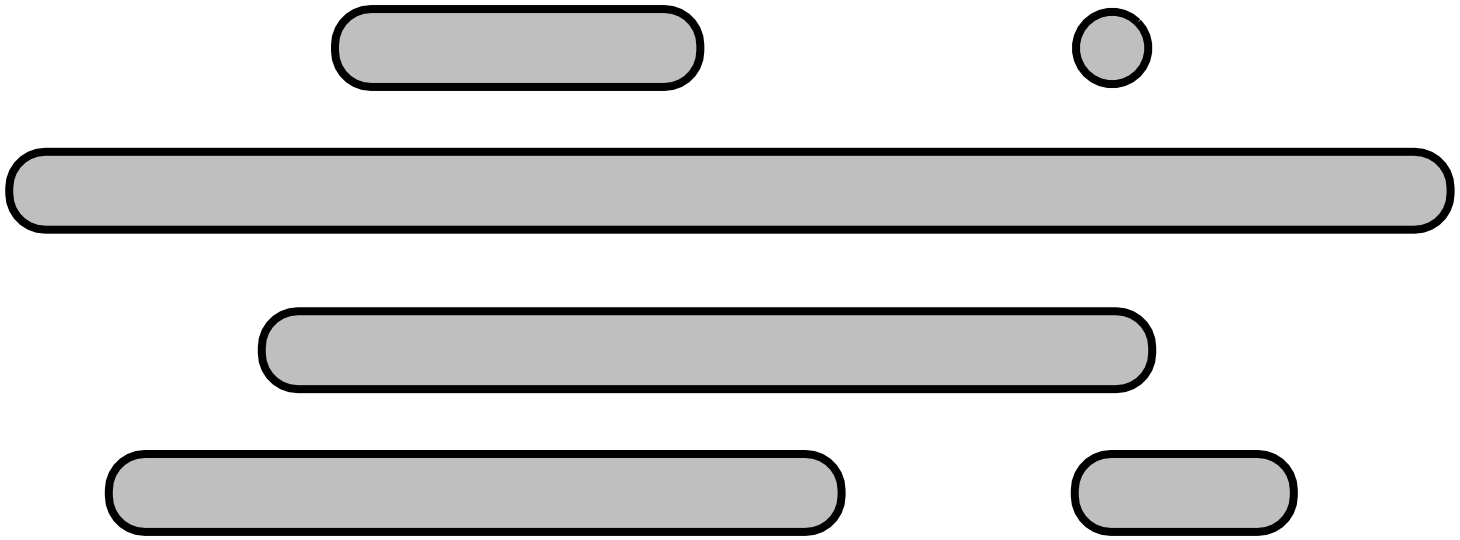


Contig Assembly



Contig Assembly

- How long are contigs?
  - Simple bacterial genomes: ~100 kbp
  - Large eukaryotic genomes: ~10-20 kbp



OLC vs DBG

- DBG
  - More sensitive to repeats and read errors
  - Graph converges at repeats of length k
  - One read error introduces k false nodes
  - Parameters: kmer_size cov_cutoff ...
- OLC
  - Less sensitive to repeats and read errors
  - Graph construction more demanding
  - Doesn't scale to voluminous short reads
  - Parameters: minOverlapLen %id ...
  - OLC assembly is best suited to lower coverage, longer read data such as Sanger, IONT, or PacBio.

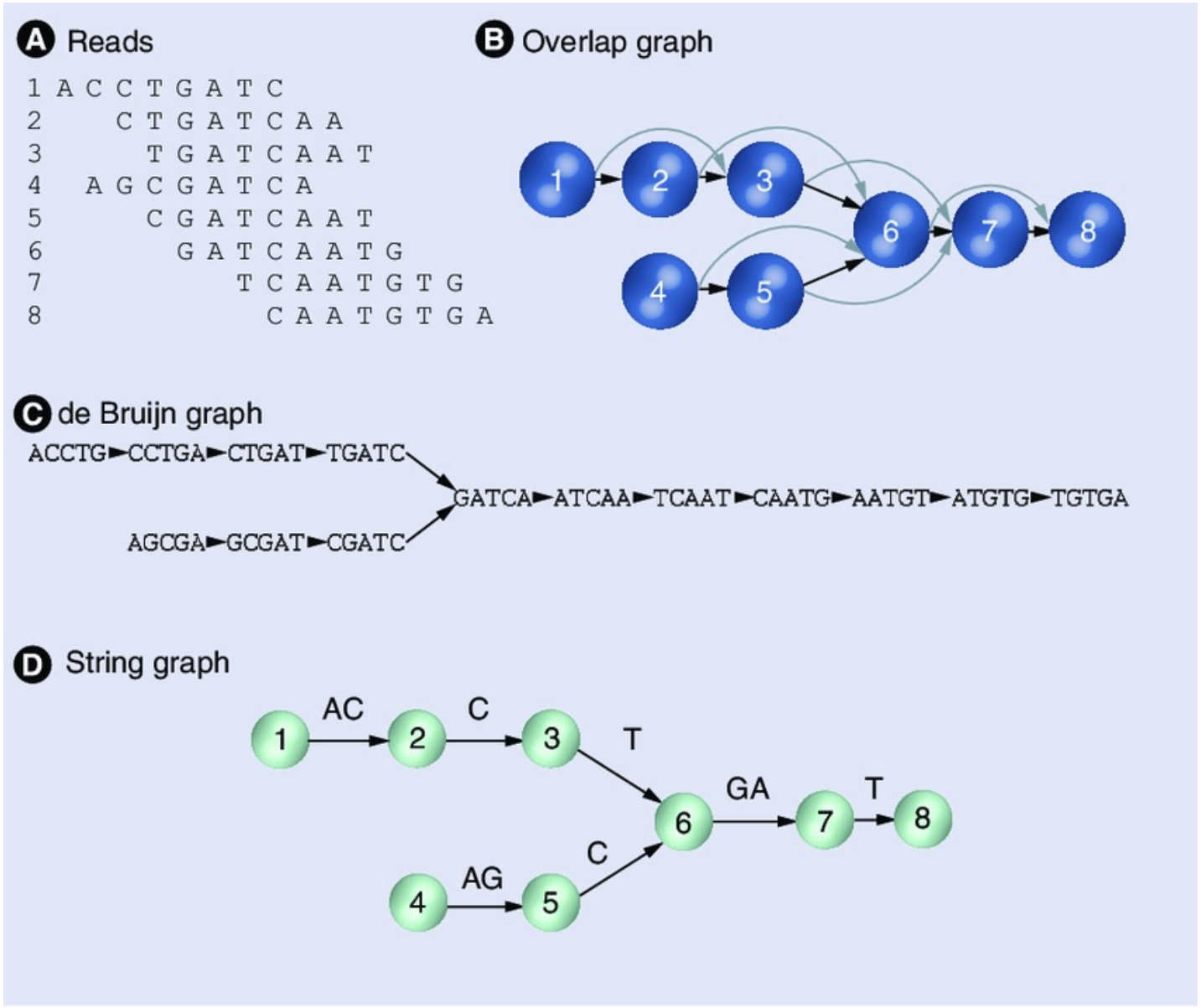
String graph assembler

The string graph is a simplified version of a classic overlap graph

- sequenced reads = NODE
- a suffix to prefix overlaps = EDGE
- the non-transitive edges = oriented graph

- The vertices in a string graph are the input reads, and the arcs correspond to the overlapping reads, which are contigs in the string graph.
- For long-read assembly, it assembles the long reads without being translated to *k*-mers.

String graph assembler

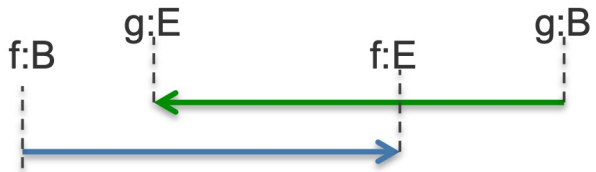


String graph assembler

For each overlap, two edges are constructed.

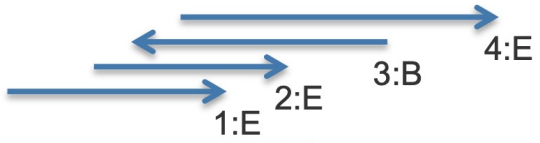
Example:

Overlapped reads

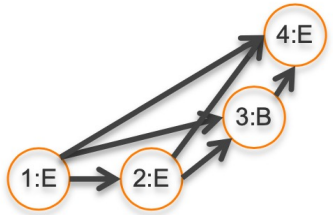


New edges

Add f:B, g:B, f:E, g:E as vertices
Add edges f:E ➔ g:B and g:E ➔ f:B

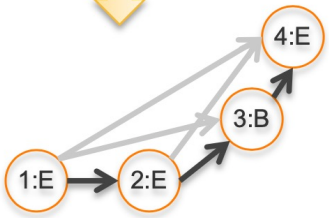


overlaps



Initial graph

Transitive Reduction



String graph

String graph assembler

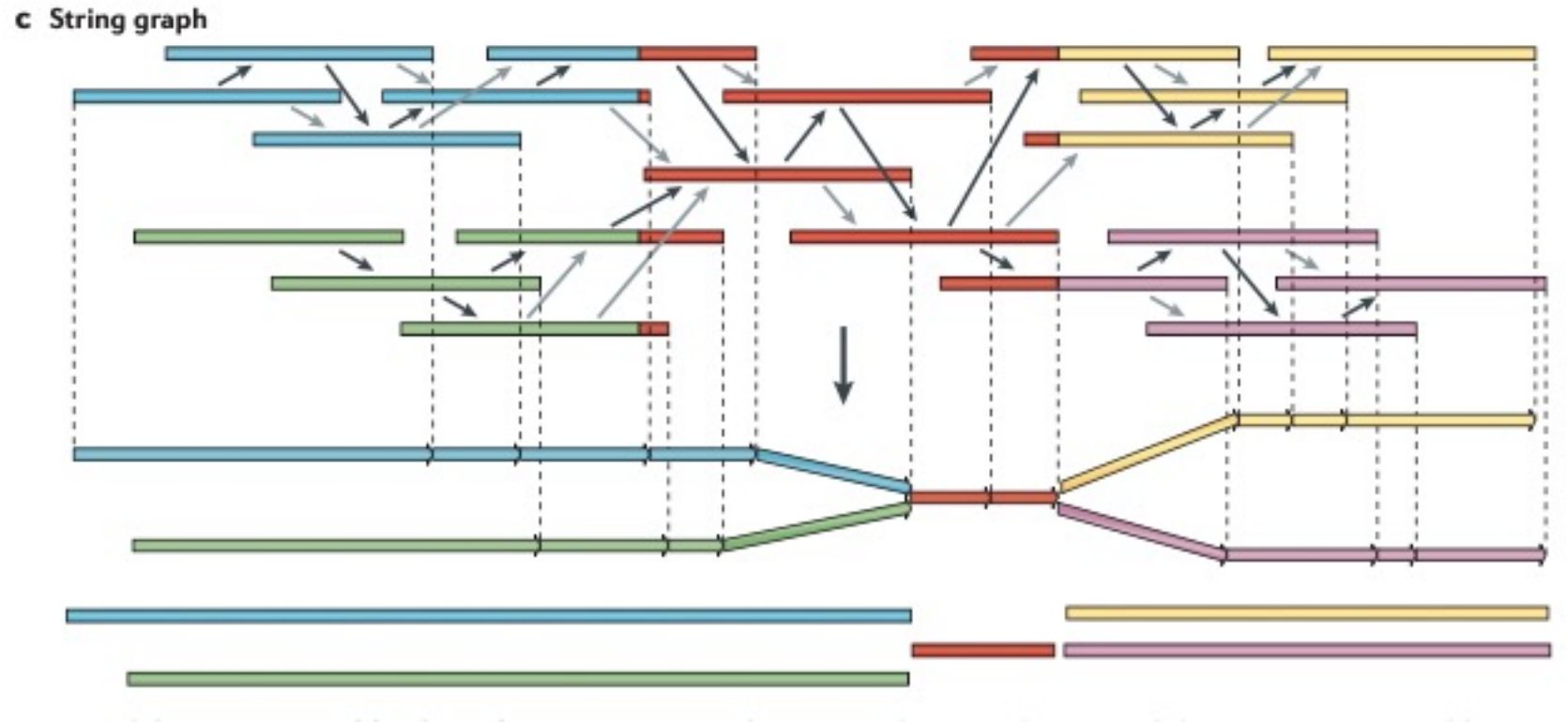
From the overlap graph, the string graph can be extracted by first

- removing duplicate reads and contained reads
- discarding transitive edges from the graph.

The formulation of the string graph assembly is similar to a de Bruijn graph in principle. However, it has the advantage of not decomposing sequences into *k*-mers, but taking the complete length of a read sequence

For long sequences and single-molecule sequencing reads with a high error rate, the overlap-based approaches are more acceptable than the de Bruijn graph-based methods.

String graph assembler



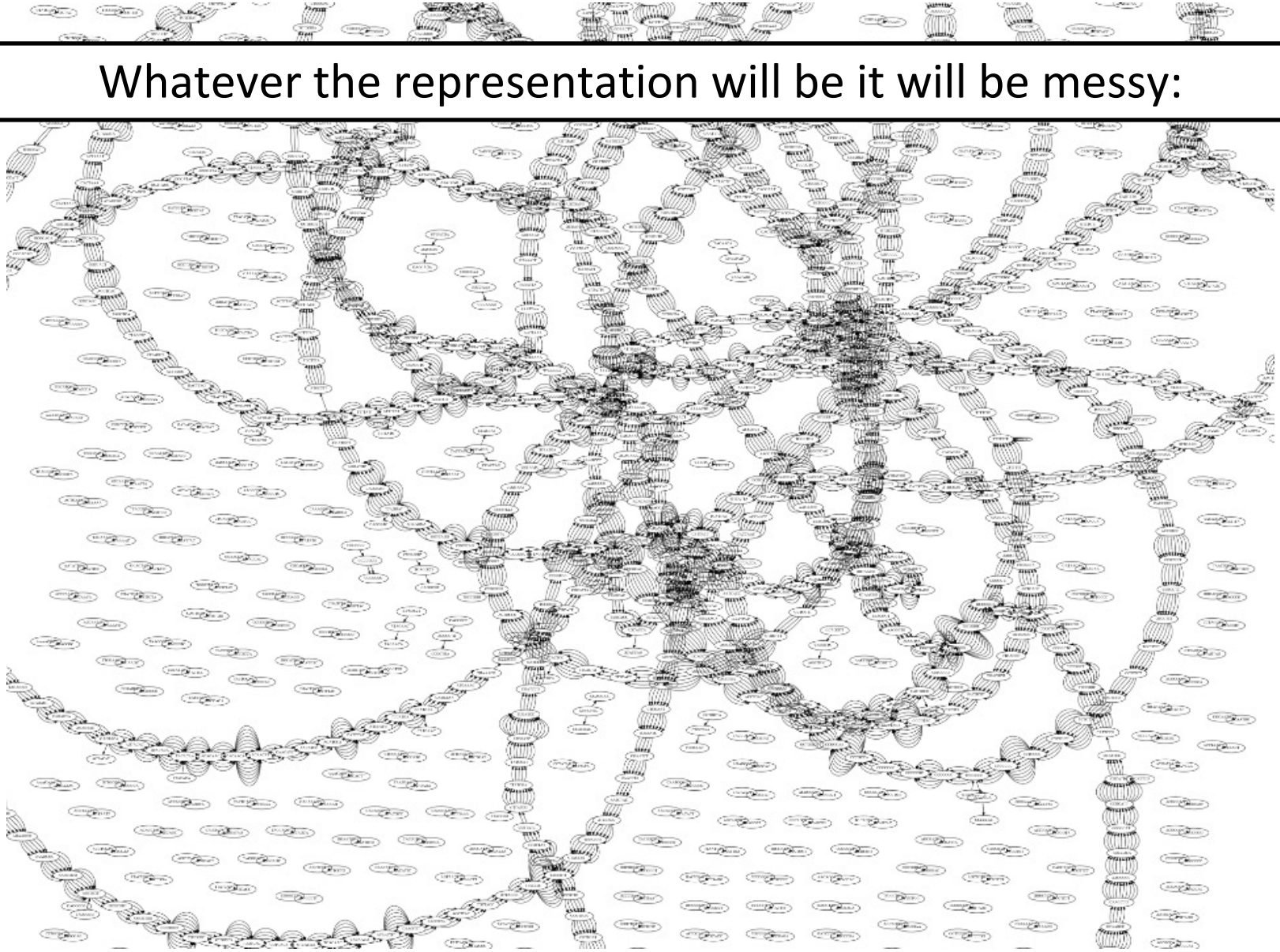
String graph. Alignments that may be transitively inferred from all pairwise alignments are removed (grey arrows). A graph is created with a vertex for the endpoint of every read. Edges are created both for each unaligned interval of a read and for each remaining pairwise overlap. Vertices connect edges that correspond to the reads that overlap. When there is allelic variation, alternative paths in the graph are formed. Not shown, but common to all three algorithms, is the use of read pairs to produce the final assembly product.

Chaisson et al. 2015 . Genetic variation and the *de novo* assembly of human genomes .doi:10.1038/nrg3933

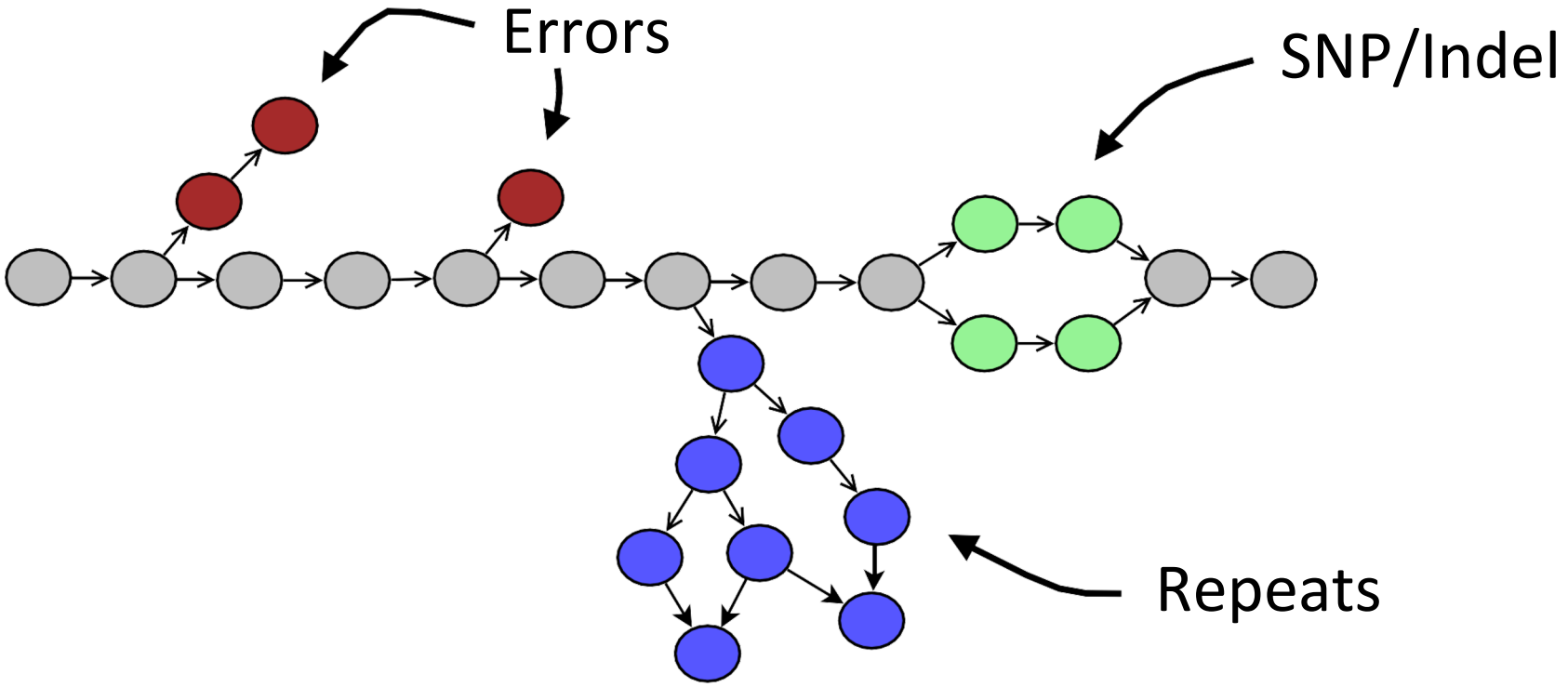
Assembly in real life

- In this topic we've learned about three ways of representing the relationship between reads derived from a genome we are trying to assemble:
- **Overlap graphs** - nodes are reads, edges are overlaps between reads.
- **De Bruijn graphs** - nodes are overlaps, edges are reads.
- **String graphs** - nodes are reads, edges are suffix to prefix overlaps

Whatever the representation will be it will be messy:



The structure of the graph



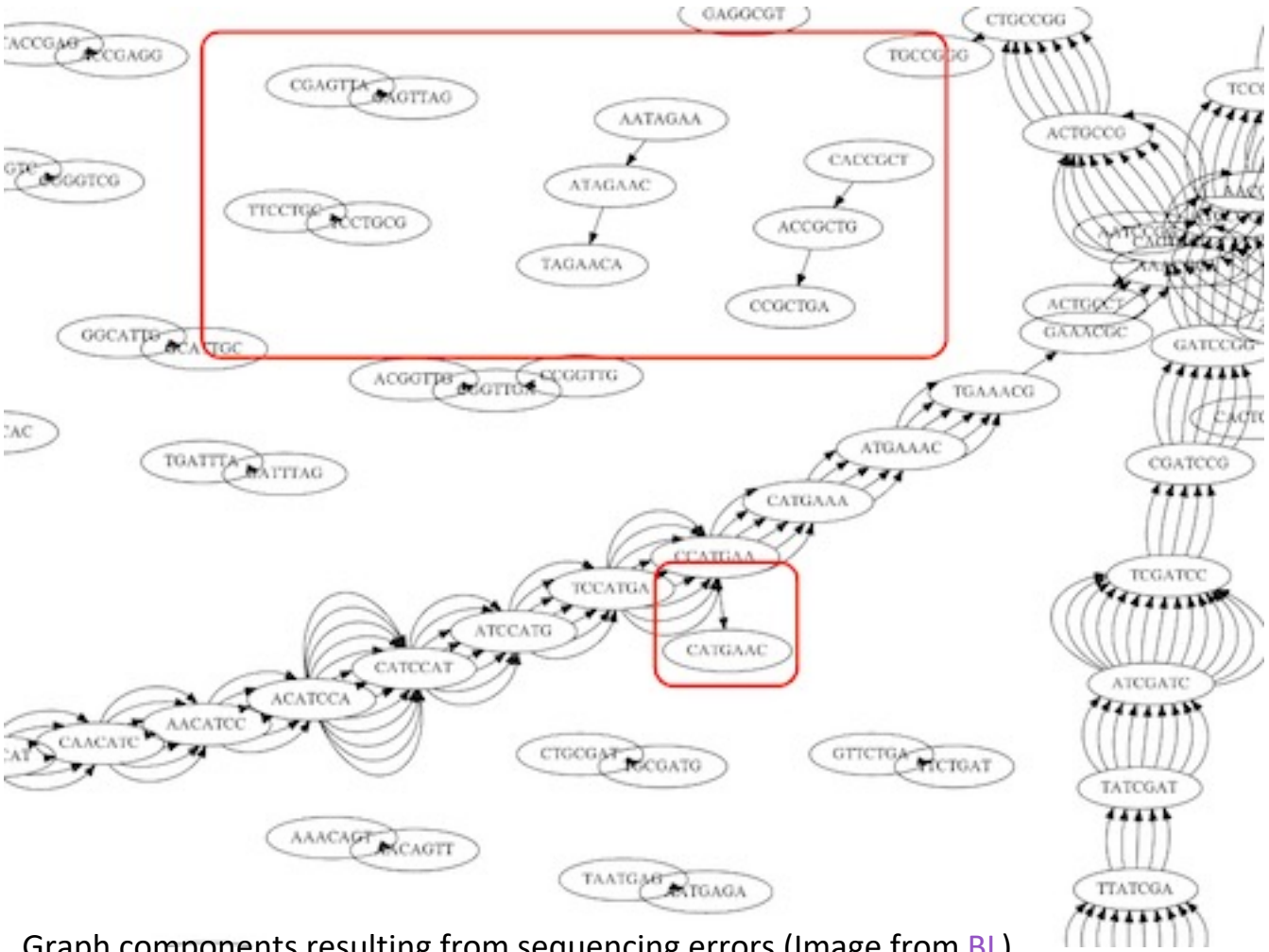
What Makes Assembly Difficult?

- Repetitive sequence
- High heterozygosity
- Low coverage
- Biased sequencing
- High error rate
- Chimeric reads
- Sequencing adapters in the reads
- Sample contamination
- Sequencing multiple individuals

There are multiple reasons for such messiness:

Sequence errors

Sequencing machines do not give perfect data. This results in spurious deviations on the graph. Some sequencing technologies such as Oxford Nanopore have very high error rate of ~15%.



Graph components resulting from sequencing errors (Image from BL).

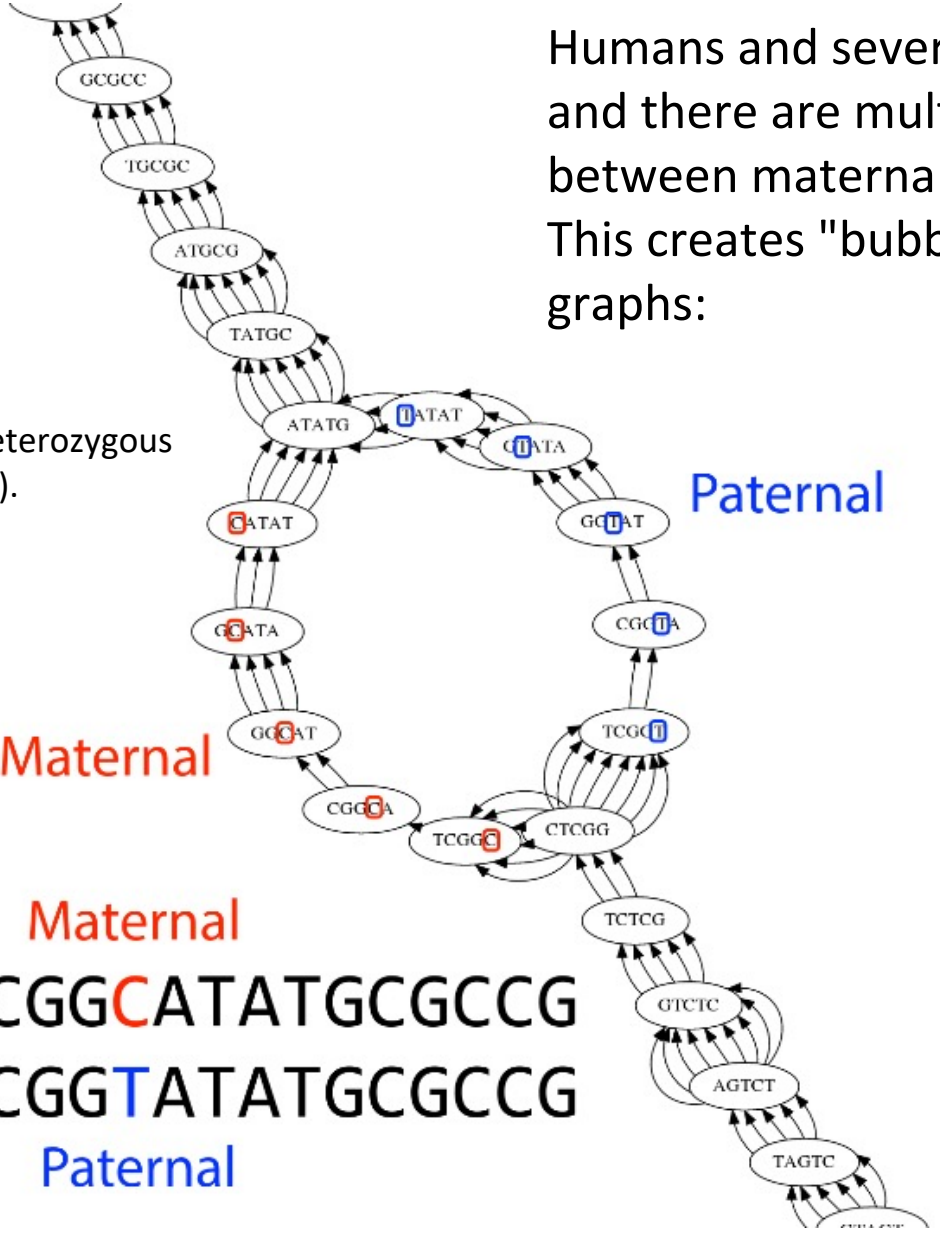
Complexity due to Genome properties

- Repeats (see later)
- GC-content
  - Regions of low or high GC-content have a lower coverage (Illumina, not PacBio)
- Secondary structure
  - Regions that are tightly bound get less coverage
- Ploidy level
  - On higher ploidy levels you potentially have more alleles present

Ploidy

Humans and several organisms are diploid and there are multiple differences between maternal and paternal genomes. This creates "bubbles" on assembly graphs:

Bubbles due to a heterozygous site (Image from BL).



Additional complexity

- Size of organism
  - Hard to extract enough DNA from small organisms
- Pooled individuals
  - Increases the variability of the DNA (more alleles)
- Inhibiting compounds
  - Lower coverage and shorter fragments
- Presence of additional genomes/contamination
  - Lower coverage of what you actually are interested in, potentially chimeric assemblies

Repetitive sequences

What is a repeat : A segment of DNA which occur more than once in the genome sequence

Very common :

- Transposons (self replicating genes)
- Satellites (repetitive adjacent patterns)
- Gene duplications (paralogs)

Repetitive sequences

Low complexity regions

Regions where some nucleotides are overrepresented, such as in homopolymers, e.g., AAAAAAAAAA, or slightly more complex, e.g., AAATAAAAAGAAAA

Tandem repeats

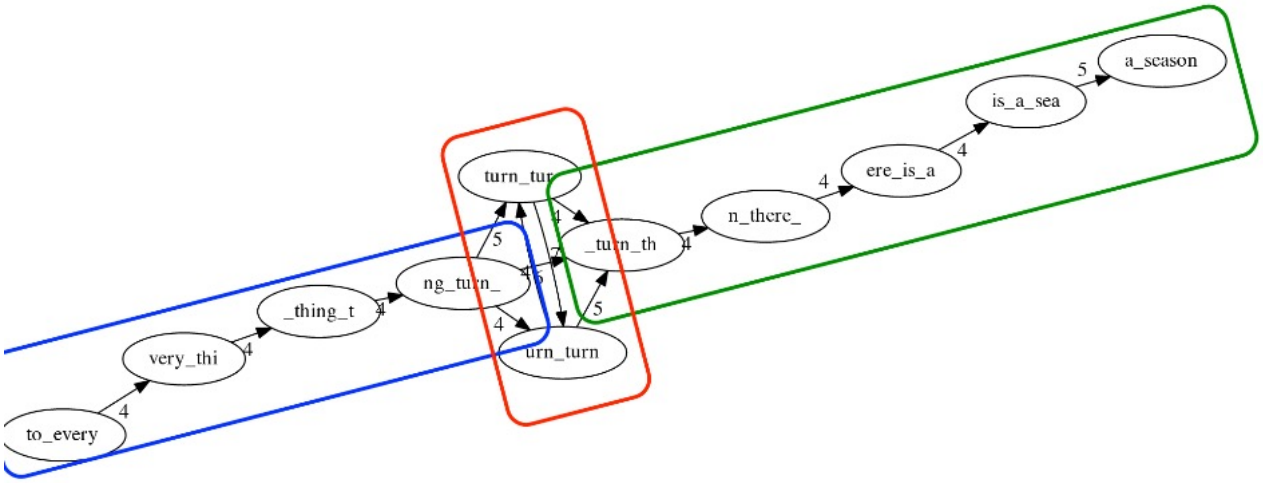
A pattern of one or more nucleotides repeated directly adjacent to each other, e.g., AGAGAGAGAGAGAGAGAGAG

- 2-5 nucleotides - microsatellites (e.g., GATAGATAGATA)
- 10-60 nucleotides – minisatellite

Complex repeats (transposons, retroviruses, segmental duplications, rDNA, etc.)

Assembling repeats

Repeats result in some regions of the genome that simply cannot be resolved and are reported in segments called *contigs*:

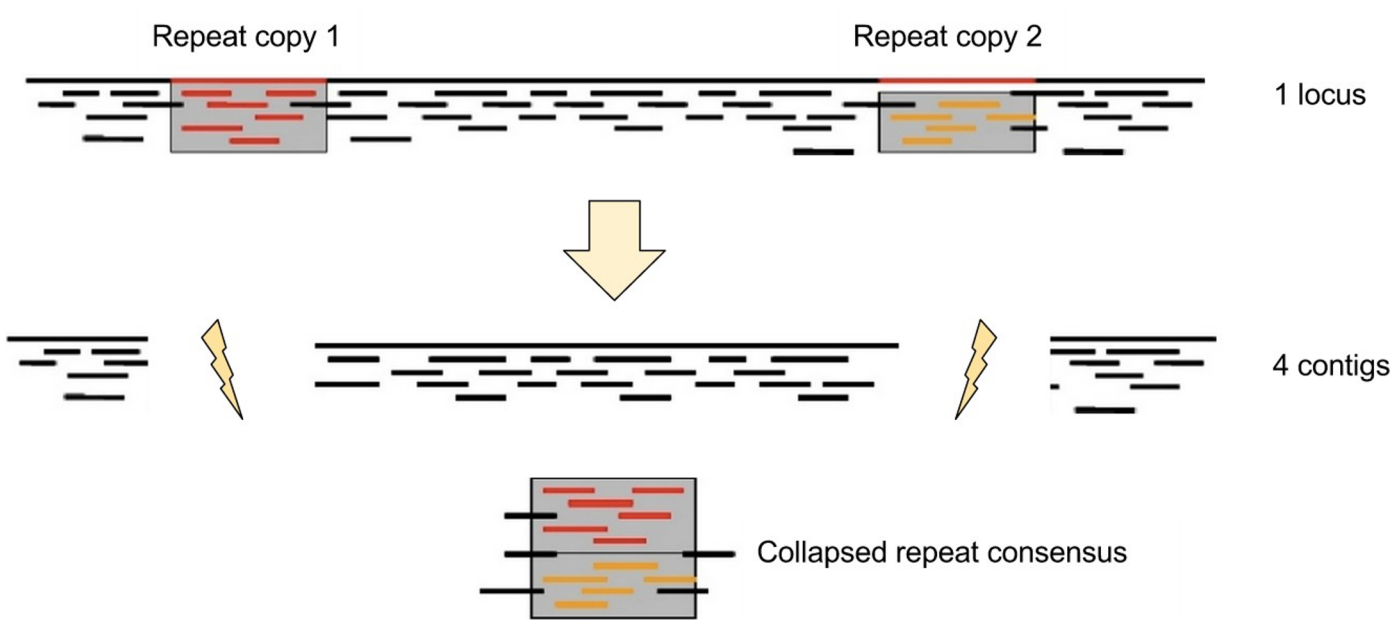


to_every_thing_turn_ _turn_there_is_a_season

_turn (repeated)

The following "genomic" segment will be reported in three pieces corresponding to regions flanking the repeat and repeat itself (Image from BL).

Assembling

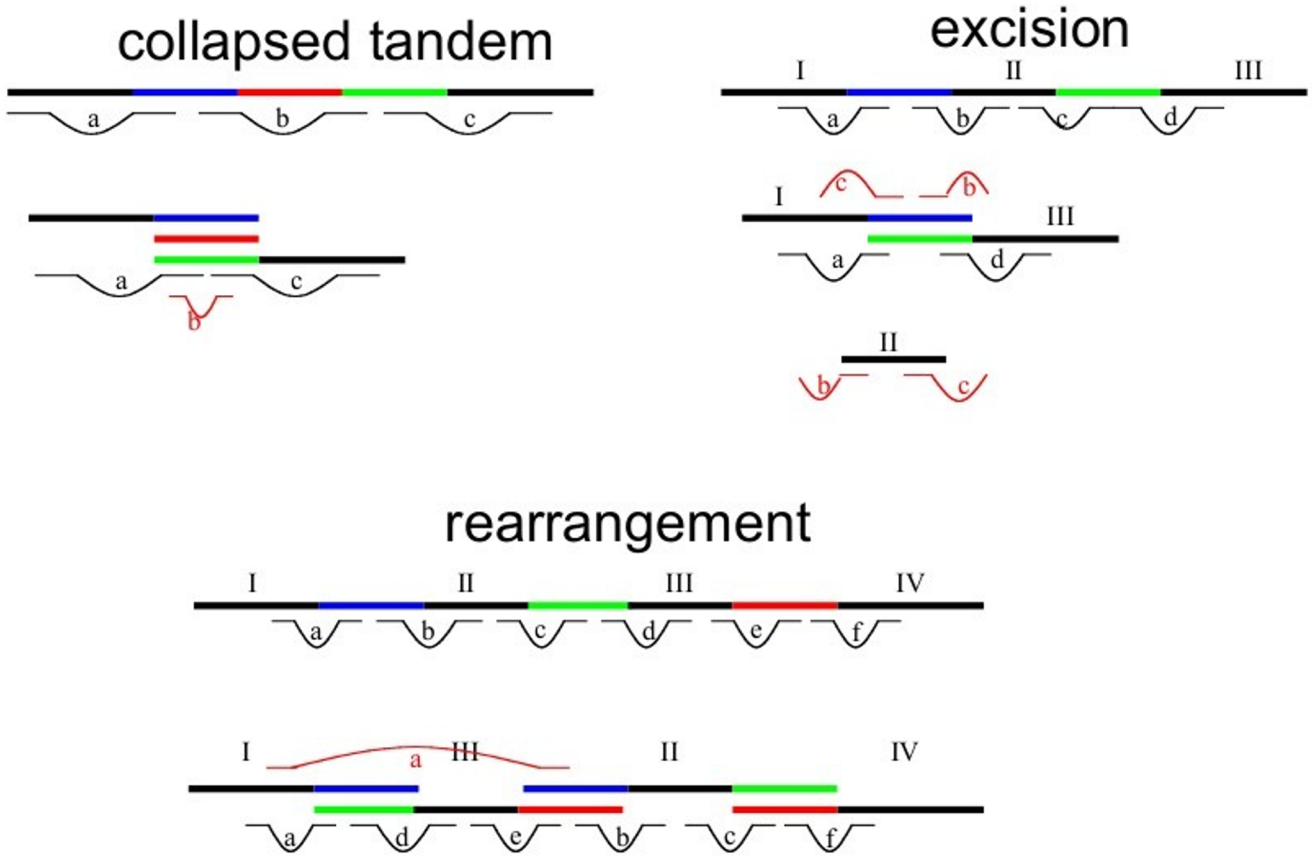


If sequencing technology produces reads > repeat size, impact is much smaller

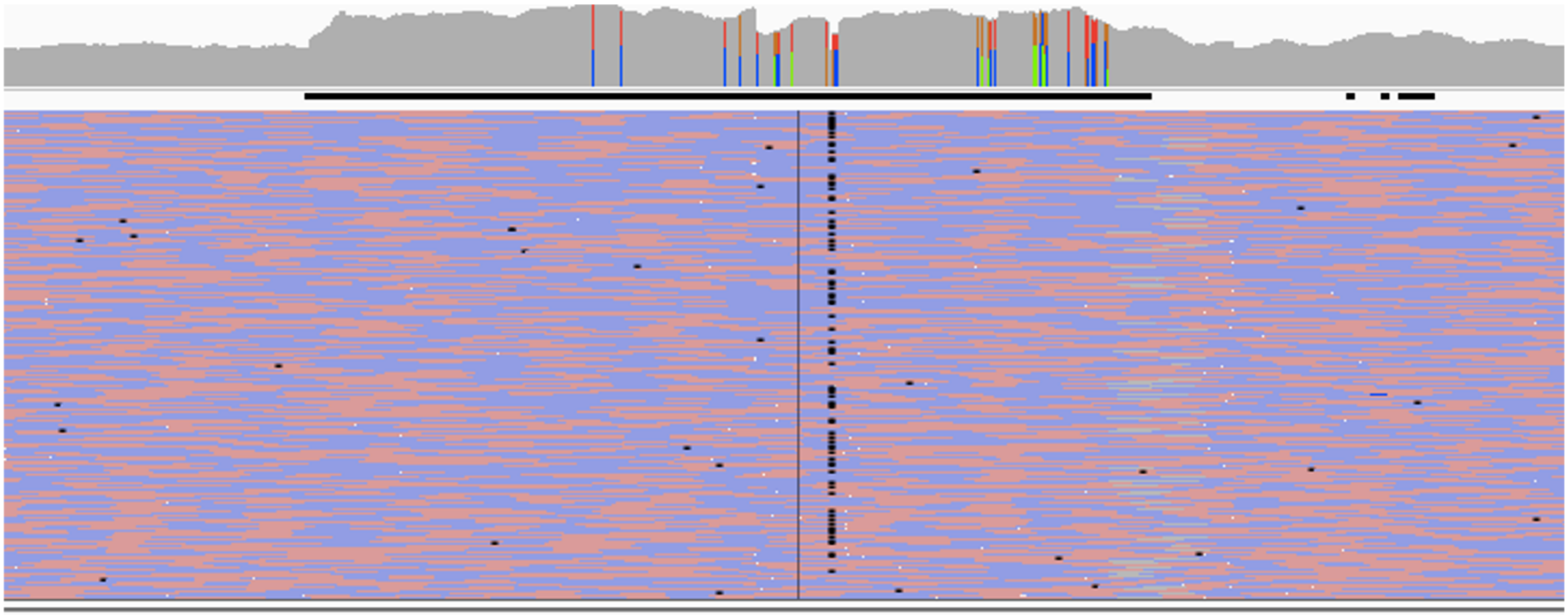
Most common solution: generate reads or mate pairs with spacing > largest known repeat

Assembling repeats

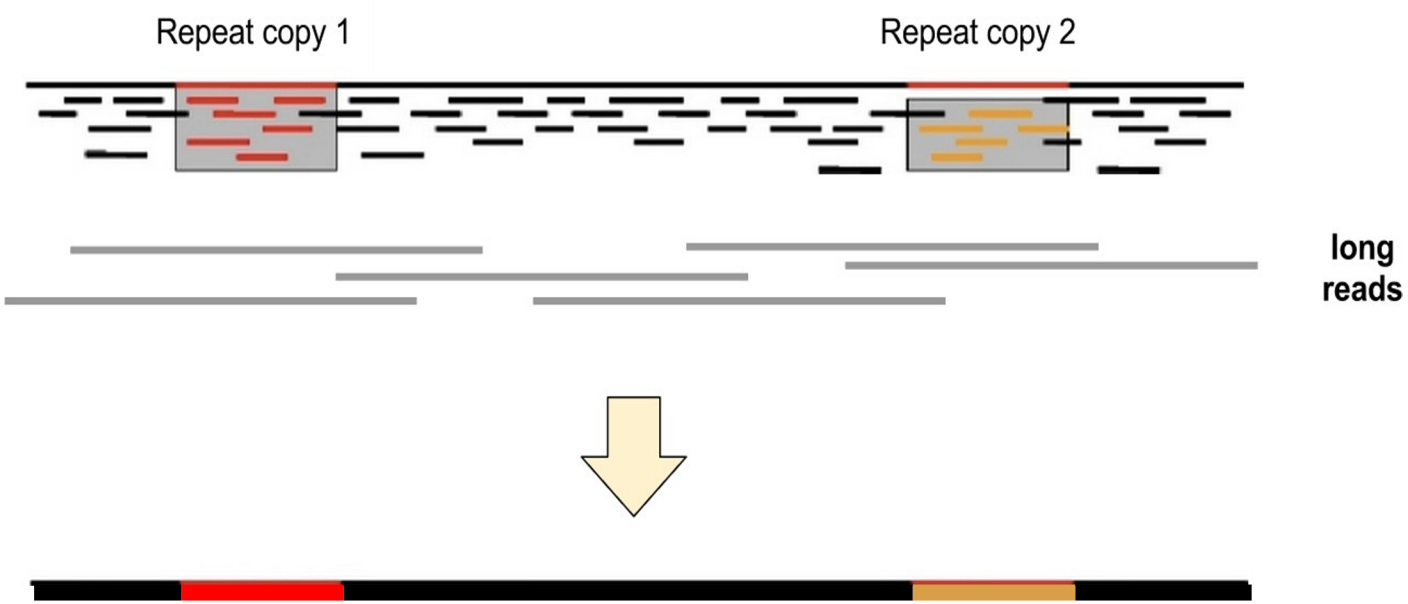
Repeat mis-assembly : Most common source of assembly errors



Assembling



Long reads



What to do with my reads?

- Short reads => DBG assemblers (e.g. Spades, ABySS, DISCOVAR, Velvet, ...)
- Long reads => OLC –SG assemblers (e.g Canu, Falcon, Hgap4, ...)
- Short + Long reads => hybrid assemblers (e.g. Unicycler, ...)
  - Hybrid assembly: long reads to resolve repeats, short reads to correct errors
- Other data and tools for polishing (scaffolding, gap filling, ...)

Few tools .

de Bruijn graphs assemblers

- SOAPdenovo, SOAPdenovo2
- SPAdes

OLC assemblers

- CANU
- HINGE
- Contagion

String graph assemblers

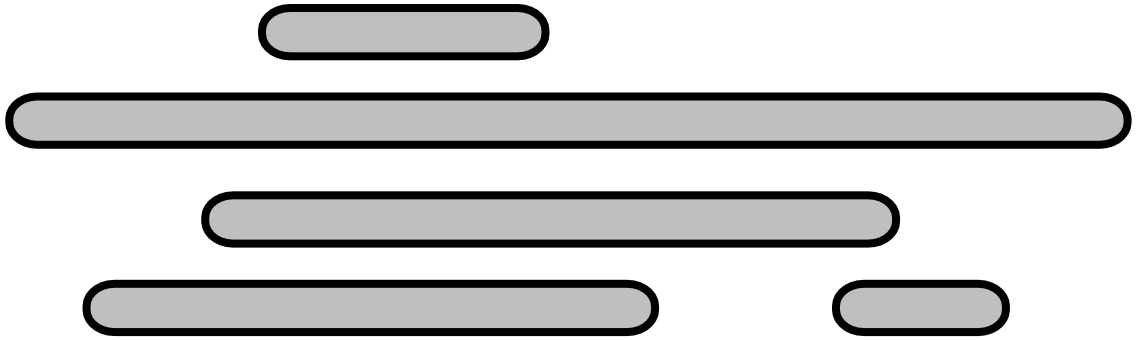
- SGA
- FALCON
- Hifiasm (HiFi pB)

Hybrid assemblers

- A5-MiSeq pipeline
- Flye
- Masurca

Scaffolding

Now, you have a huge pile of contigs but you want to make them larger.



How? -> Add context!

Essentially, link together contigs using other genomic information (longers reads f.e.) Where contigs might be located on the genome relative to one another

Scaffolding : DNA sequencing Tricks

Illumina sequencing (100 to 300 bp reads)

Paired-end reads  <500bp fragment

Mate pair reads  >5kb fragment

>250 bp PCRfree reads 

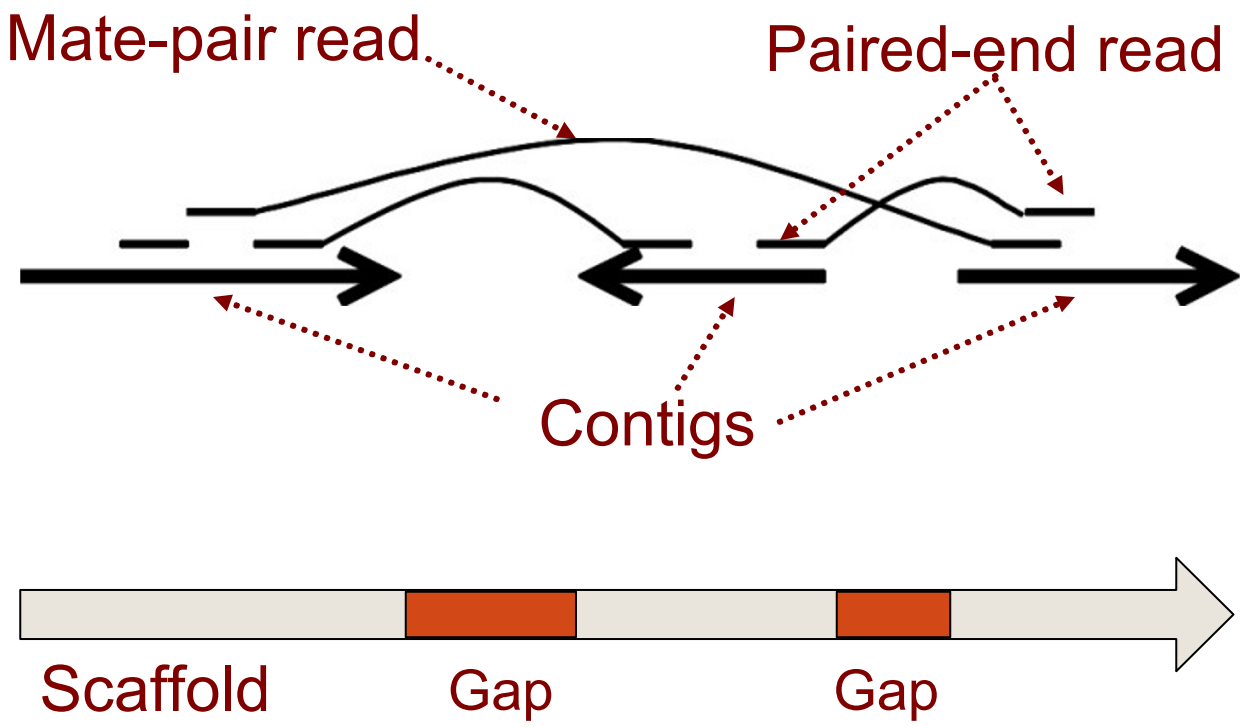
10x Genomics Linked reads  >50kb fragments

PacBio/ONT long-read (10 – 15+ kb reads)

10-15 kb 

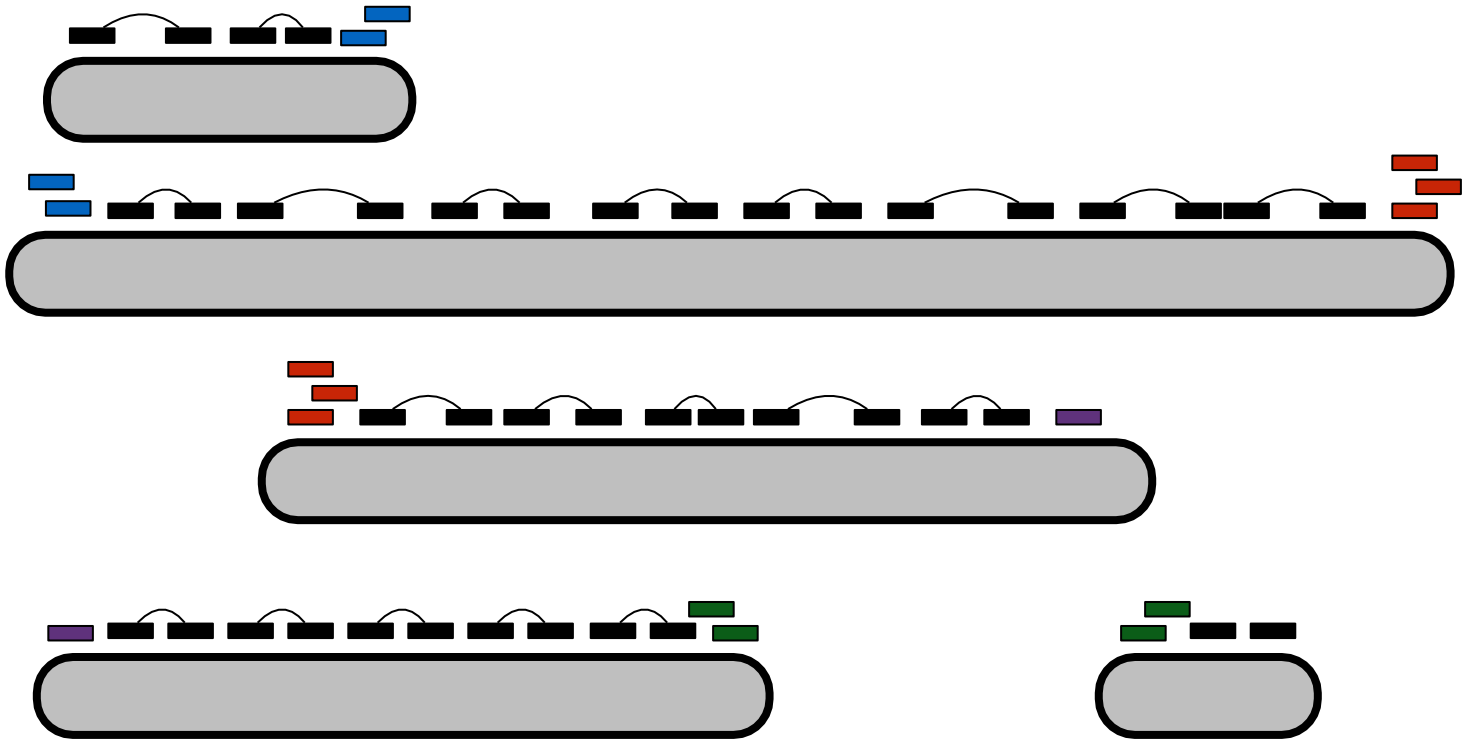
Modified from Qi Sun, Minghui Wang, Cornell Univ.

Contigs to scaffolds



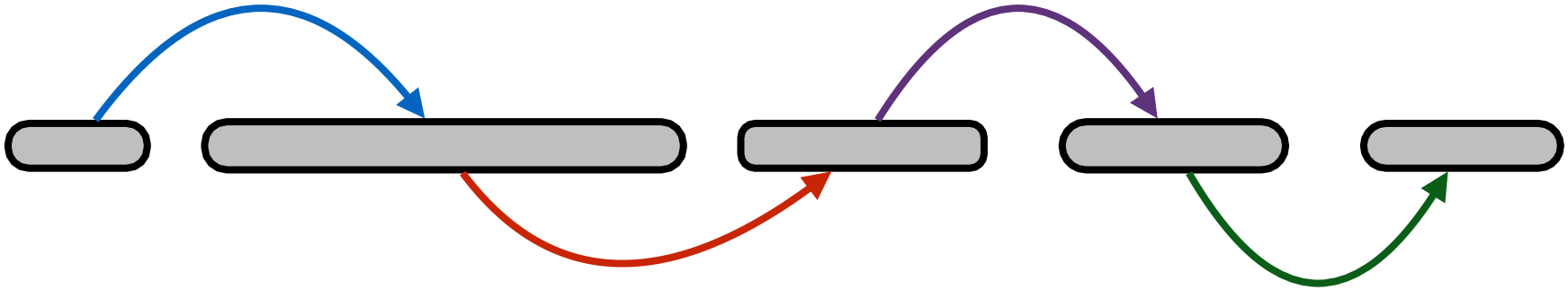
Scaffolding

- Scaffolding starts by mapping read pairs to contigs



Scaffolding

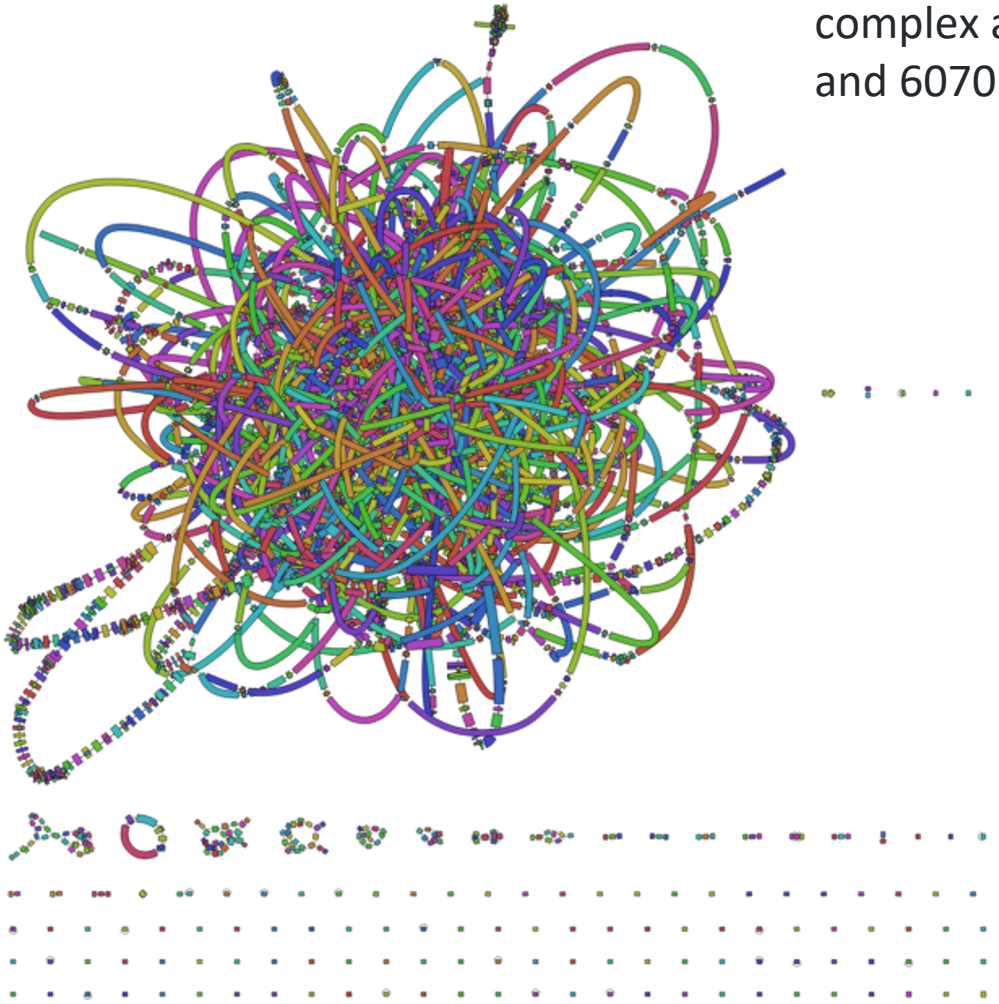
- Read pairs help us build a scaffold graph
- Estimate distances between contigs using fragment size distribution



Gap Filling

- Scaffolds contain gaps ('NNNNNN')
  - Can use local assembly to fill these in
  - Examples: sga gapfill, GapCloser from SOAPdenovo
- Can fill gaps using other sequencing technology (eg PacBio)

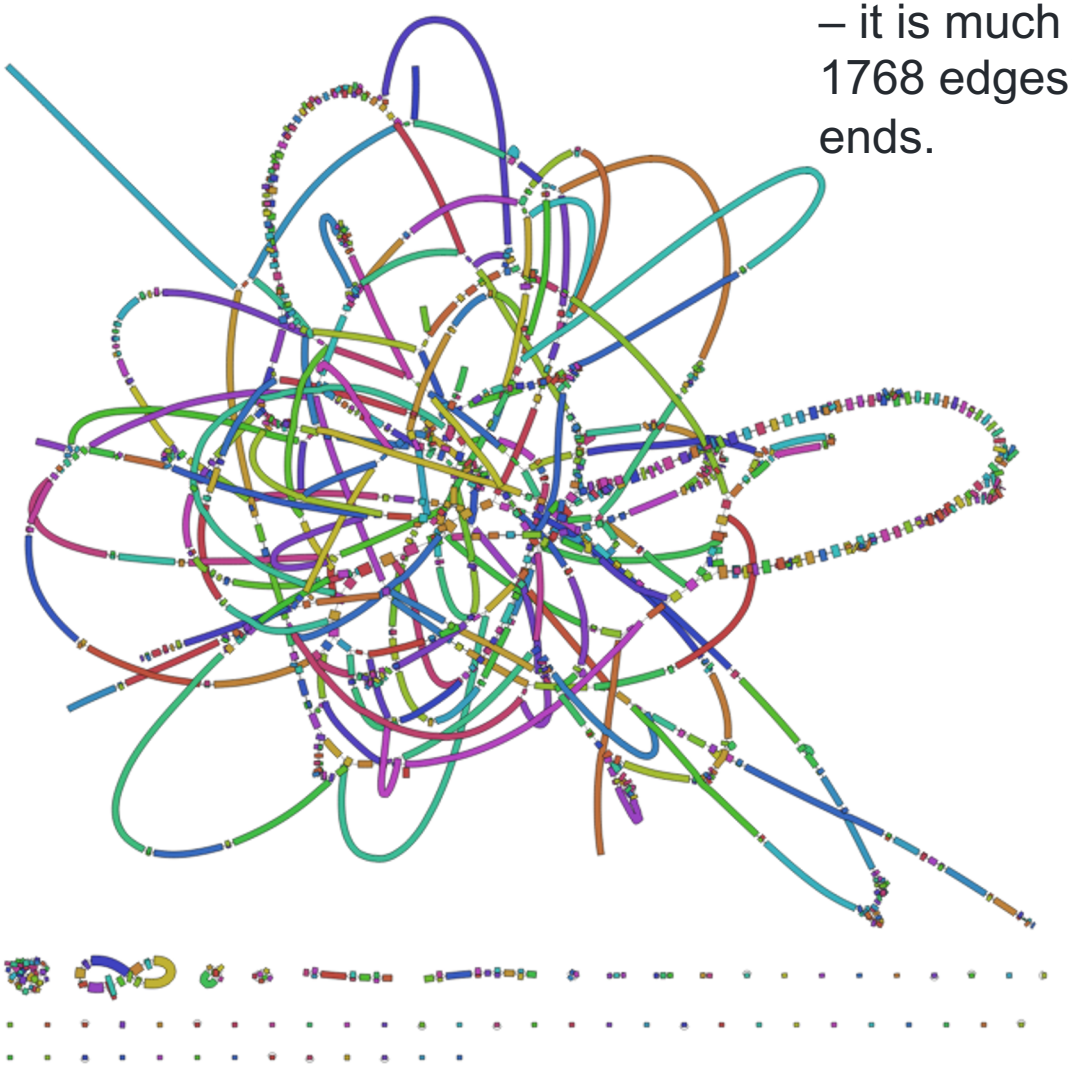
K-mer size effect



K- mer : 51

This k-mer size is too small, resulting in a complex and tangled graph with 4618 nodes and 6070 edges.

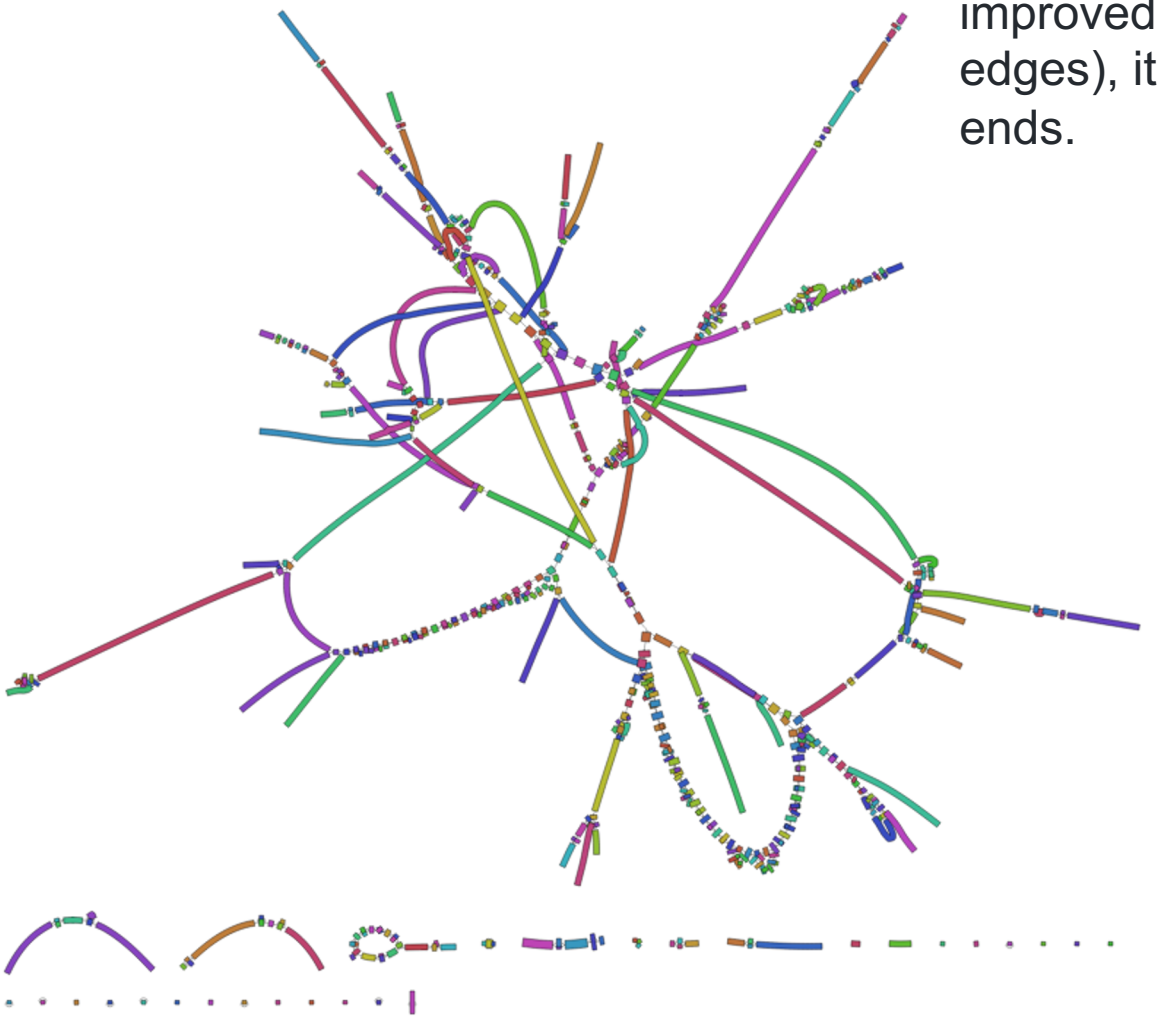
K-mer size effect



K- mer : 61

This graph is better than the 51-mer graph – it is much less complex (1357 nodes and 1768 edges) but still has very few dead ends.

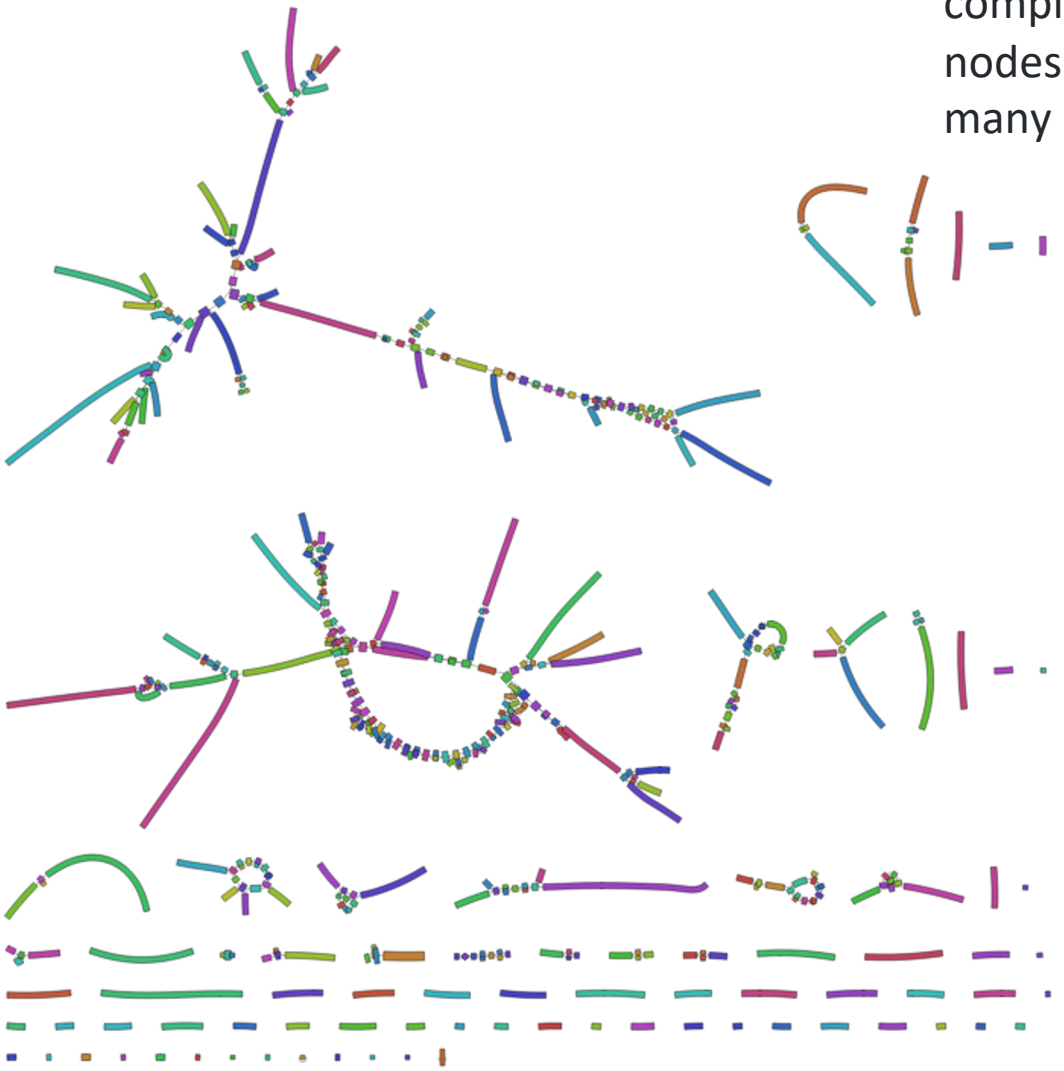
K-mer size effect



K- mer : 71

While the complexity of the graph has improved (it has 611 nodes and 765 edges), it now shows many more dead ends.

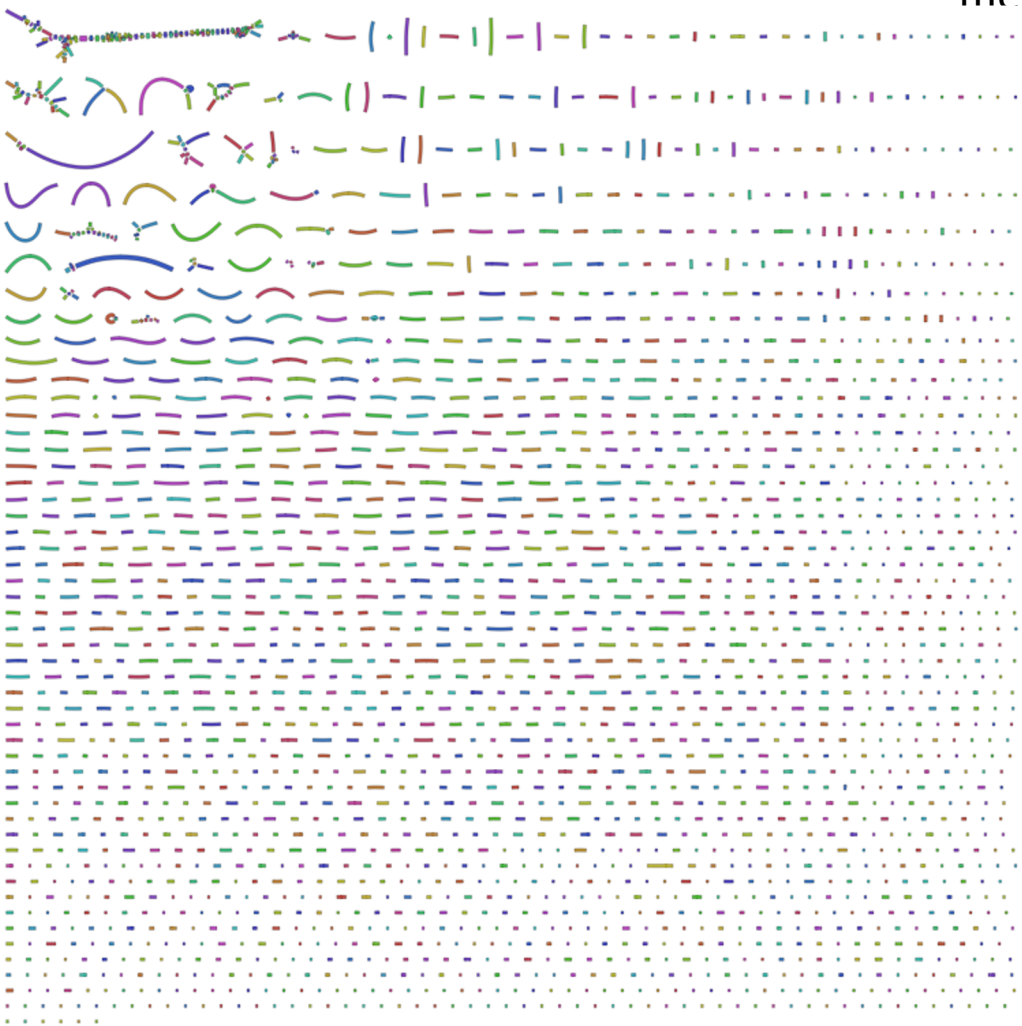
K-mer size effect



K-mer : 81

As compared to the 71-mer graph, the complexity has slightly improved (it has 490 nodes and 512 edges), but it has broken into many disconnected parts.

K-mer size effect



K-mer : 91

This graph has 2386 nodes and 304 edges and mostly consists of disconnected nodes. This k-mer size is definitely too large.

K-mer size effect

SPAdes like other assembler doesn't use a single k-mer size per assembly but rather **a range of k-mer sizes**, where each subsequent graph is built on the previous one.

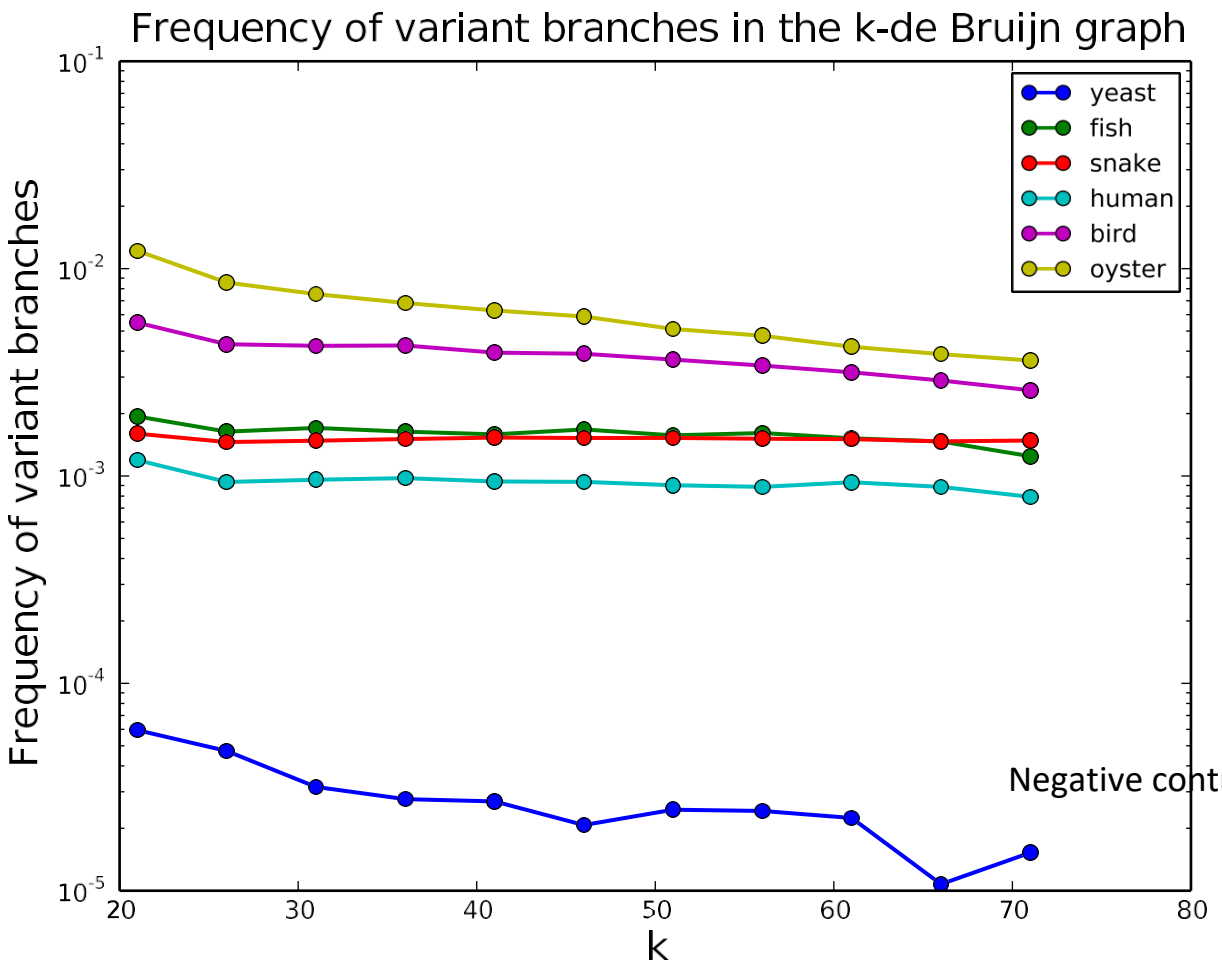
The result is that SPAdes **graphs are less prone to breaking apart at high k-mers** than Velvet graphs. But even in SPAdes, k-mer ranges that go too high can result in less ideal assemblies.

A maximum k-mer size of **about 80% of the read length** may yield good results in SPAdes. E.g. if assembling 100 bp reads, a k-mer range of 21,33,55,77 may work well.

If assembling somewhat longer reads (such as 300 bp MiSeq reads), you can try going all the way to the SPAdes maximum k-mer of 127.

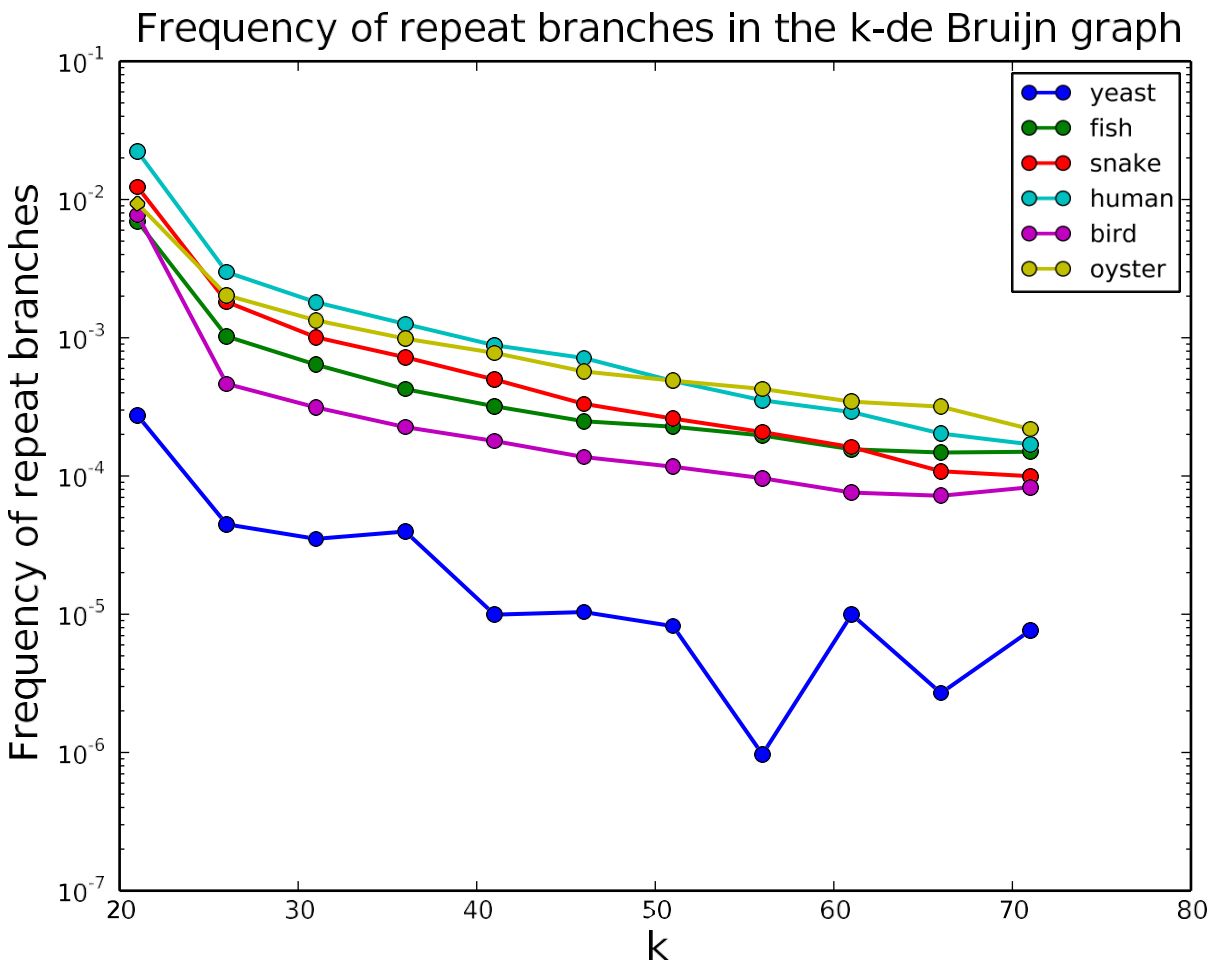
https://github.com/rrwick/Bandage/wiki/Effect-of-kmer-size

K-mer size effect : Variant branche rate

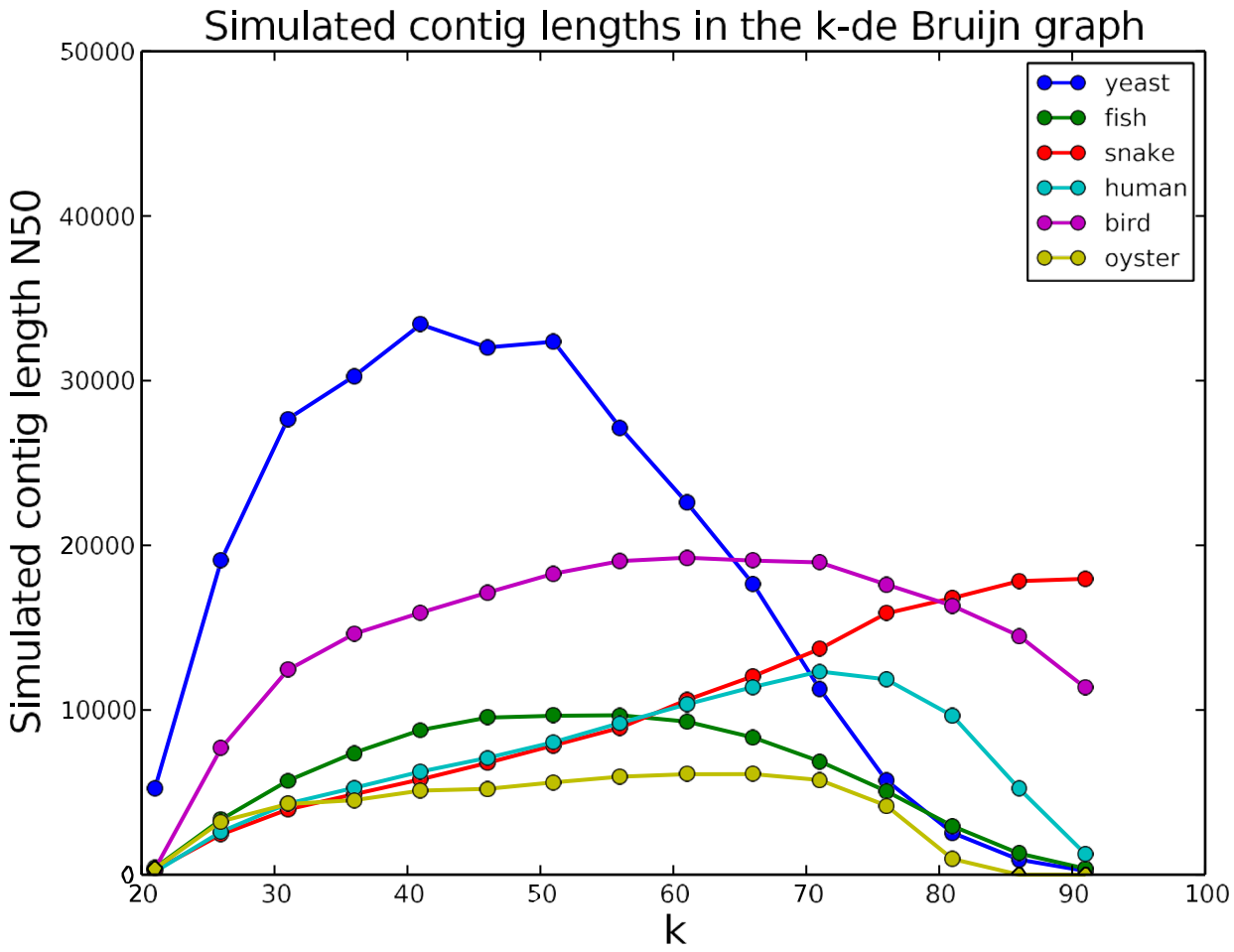


Negative control : haploid genome

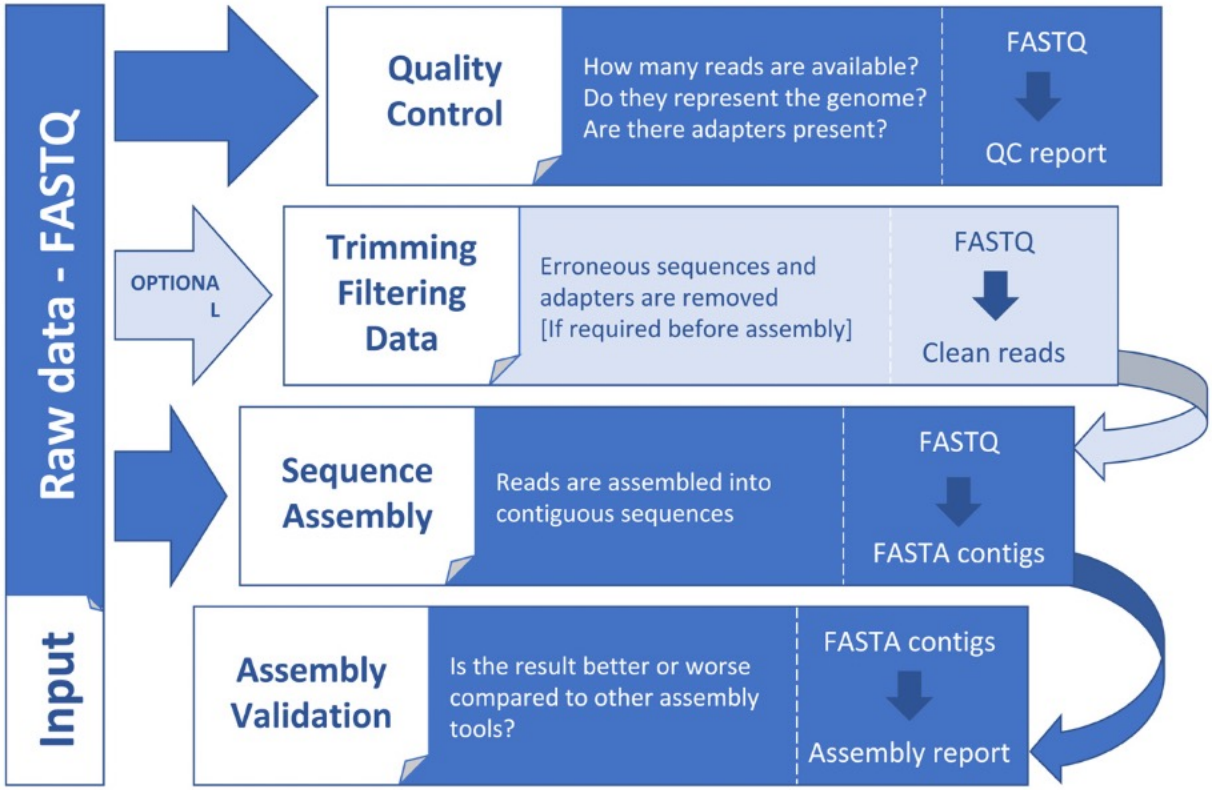
K-mer size effect : Repeat branche rate



K-mer size effect : simulated contigs length

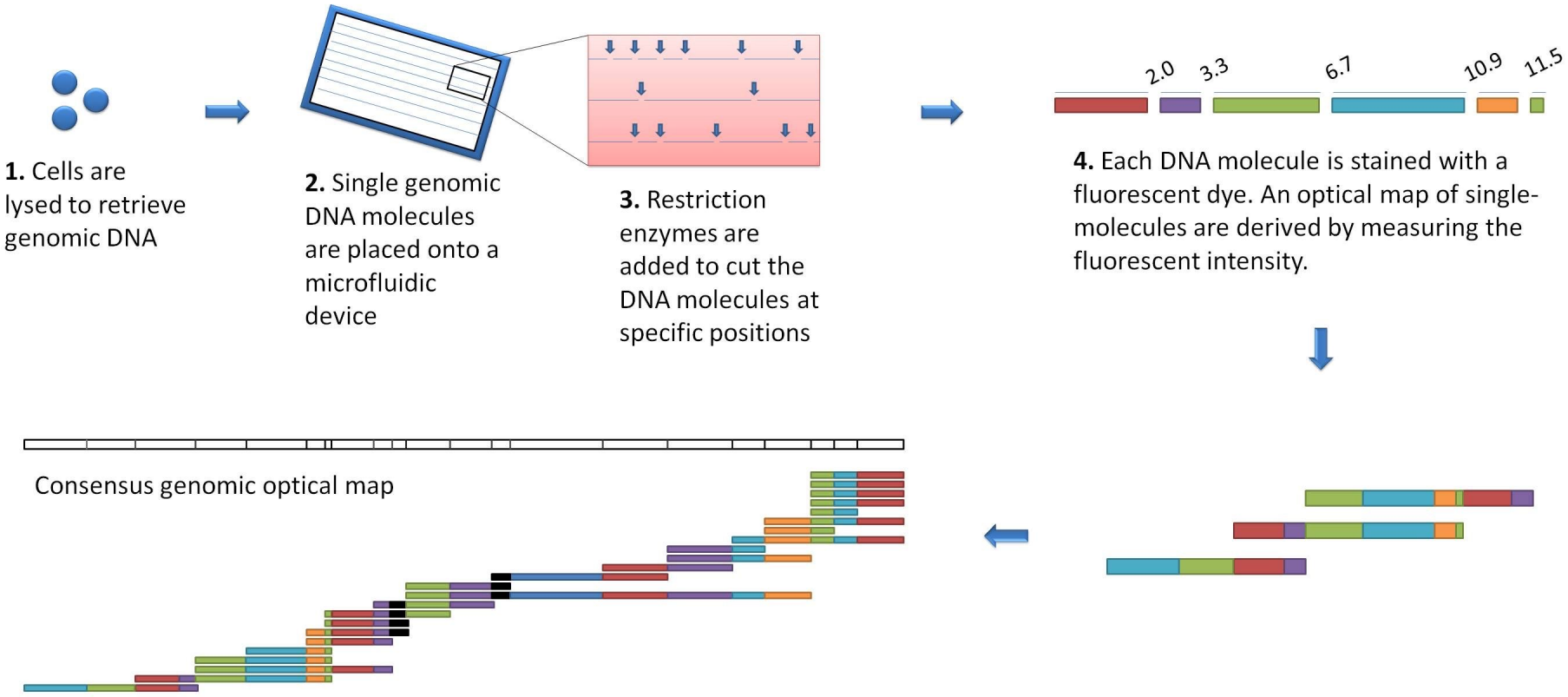


In resume



1.Angel, V. D. D. *et al.* Ten steps to get started in Genome Assembly and Annotation. *F1000research* **7**, ELIXIR-148 (2018).

Other data: Optical maps



1. Cells are lysed to retrieve genomic DNA

2. Single genomic DNA molecules are placed onto a microfluidic device

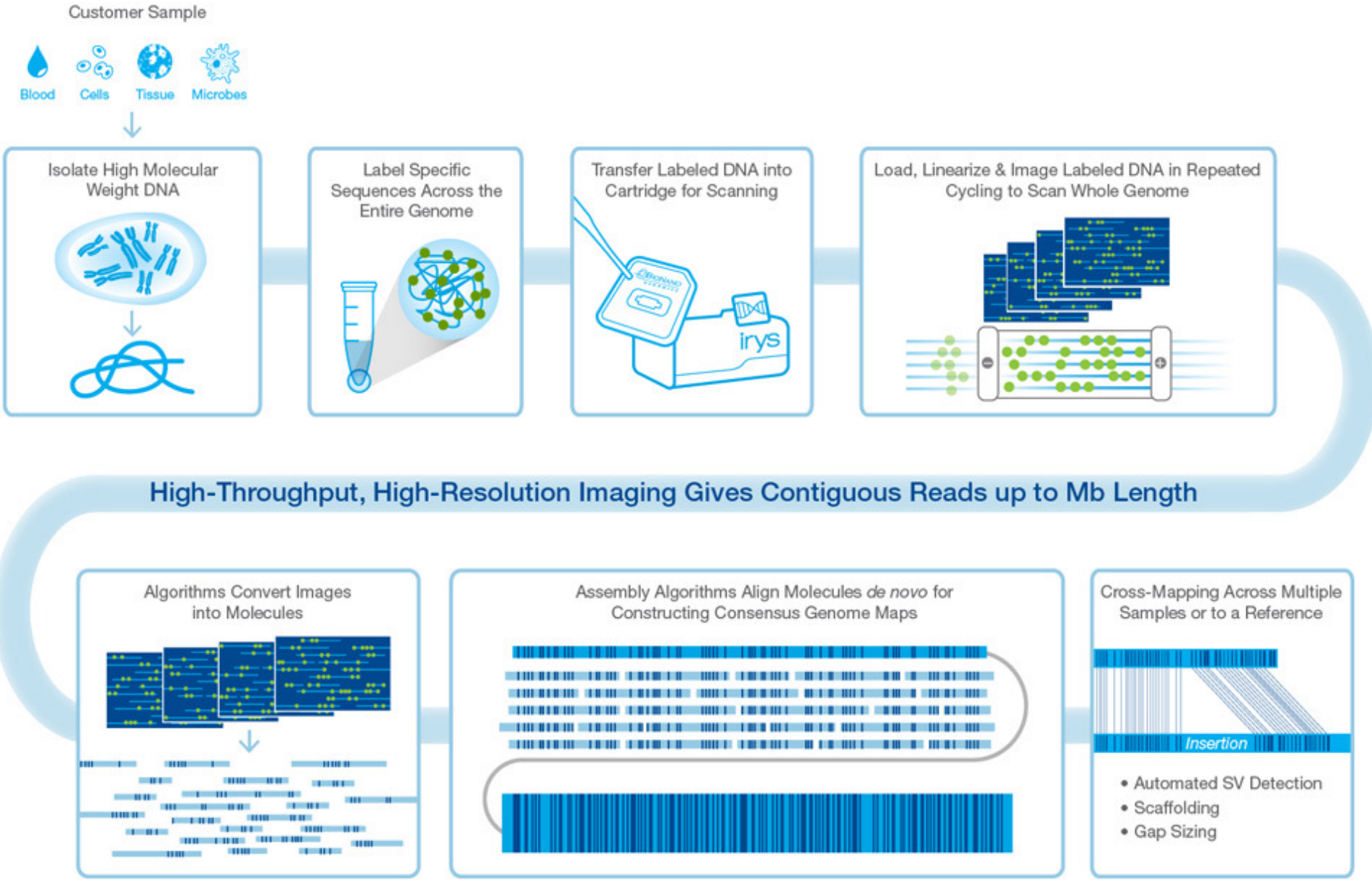
3. Restriction enzymes are added to cut the DNA molecules at specific positions

4. Each DNA molecule is stained with a fluorescent dye. An optical map of single-molecules are derived by measuring the fluorescent intensity.

5. Overlapping of the multiple single-molecule maps gives us the consensus genomic optical map

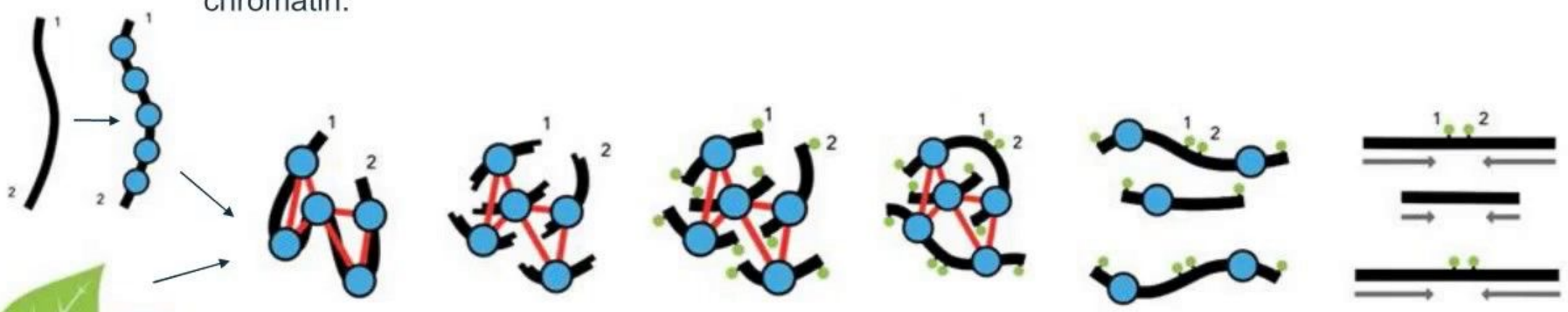
Use for scaffolding by comparing the restriction map with the predicted restriction sites in the contig sequences

BioNano


Customer Sample
Blood
Cells
Tissue
Microbes
Isolate High Molecular Weight DNA
Label Specific Sequences Across the Entire Genome
Transfer Labeled DNA into Cartridge for Scanning
irys
Load, Linearize & Image Labeled DNA in Repeated Cycling to Scan Whole Genome
High-Throughput, High-Resolution Imaging Gives Contiguous Reads up to Mb Length
Algorithms Convert Images into Molecules
Assembly Algorithms Align Molecules de novo for Constructing Consensus Genome Maps
Cross-Mapping Across Multiple Samples or to a Reference
Insertion
• Automated SV Detection
• Scaffolding
• Gap Sizing

Proximity Ligation Approaches

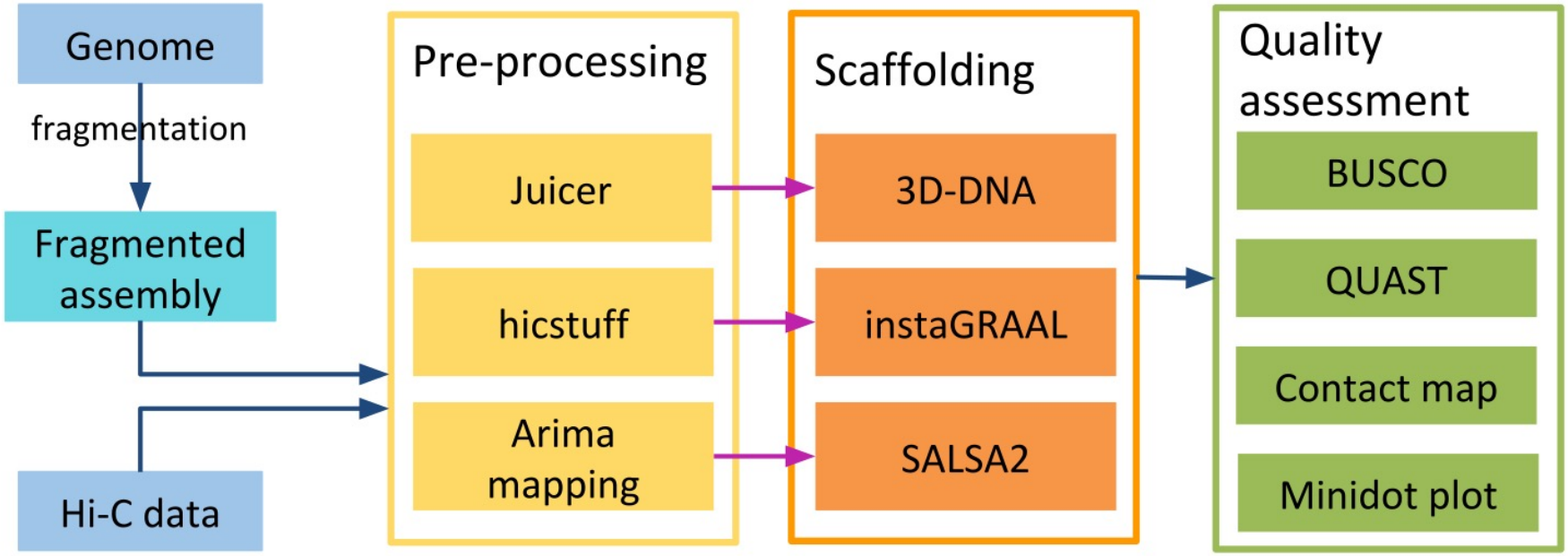
Chicago™ libraries start from pure DNA that is reconstituted into chromatin.



Dovetail™ Hi-C libraries start from tissue or cell-culture and endogenous chromatin is extracted after fixation.

HiRise™ Scaffolding Pipeline

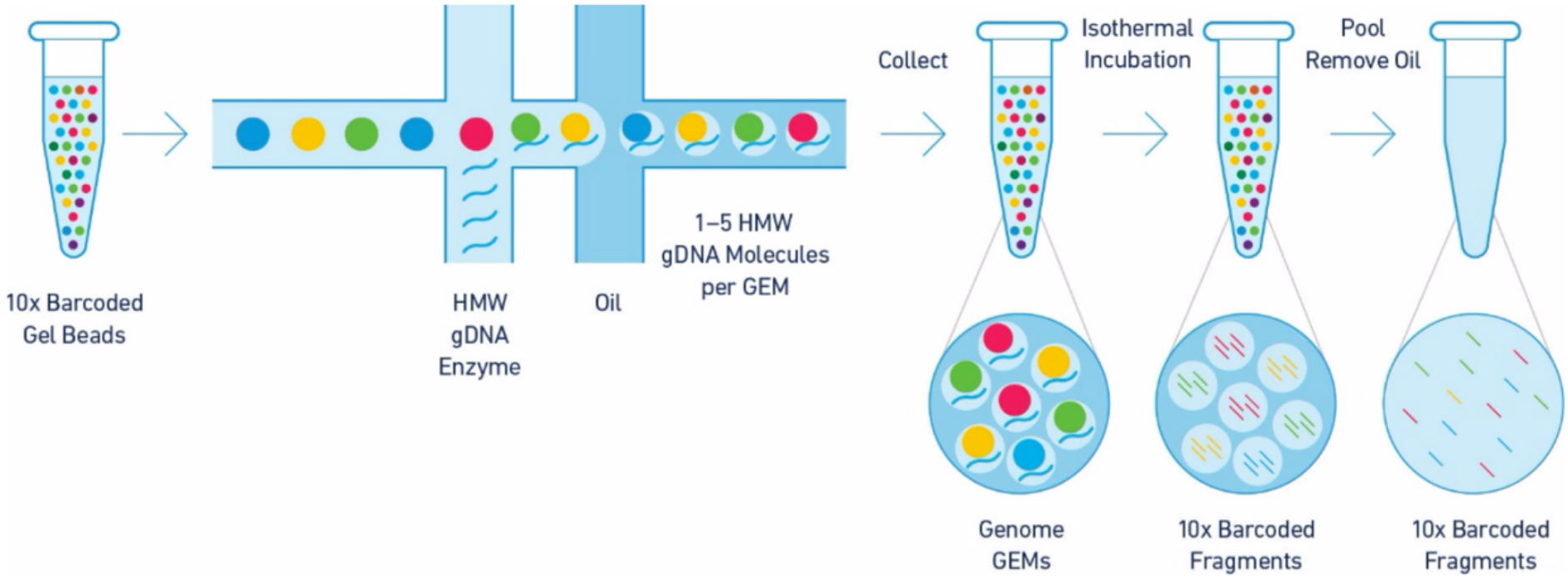
Hi-C


Genome
fragmentation
Fragmented assembly
Hi-C data
Pre-processing
Juicer
hicstuff
Arima mapping
Scaffolding
3D-DNA
instaGRAAL
SALSA2
Quality assessment
BUSCO
QUAST
Contact map
Minidot plot

Other data: Linked-Reads

- Long DNA fragments are separated in gel beads (gems) and then sequenced with Illumina HiSeq -> a “cloud” of reads originating from the same (long) DNA fragment
- These reads can then be used to assemble the genome (Supernova) or scaffold/phase the genome (Architect)

10x genomics



Reads grouped by physical regions on the genome thanks to barcodes
10X Genomics (discontinued in 2020)