Bioinformatics for Evolutionary Biology

Assembly: algorithms and tools

What is an assembly

- Given: A set of reads (strings) $\{s_1, s_2, \ldots , s_n\}$
- Do: Determine a superstring ***s*** that “best explains” the reads

- What do we mean by “*best explains*”?

Typical assembly

Multiple copies of sample DNA

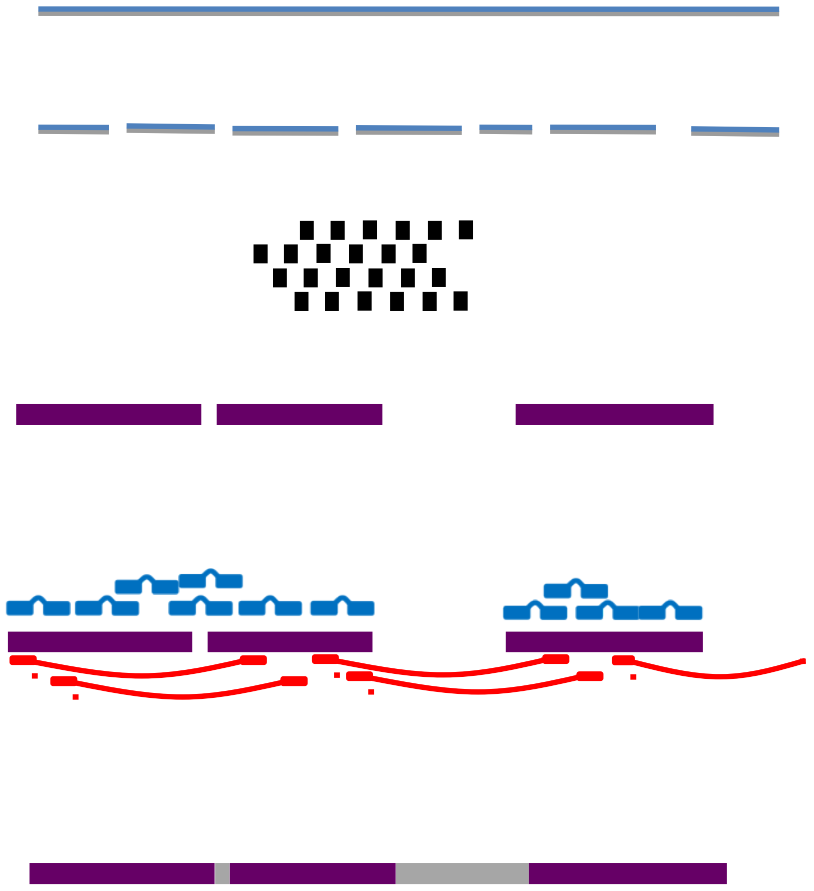
Randomly fragment DNA

Reads

Contigs

Scaffolds

Assembly



Size matters

- Small (e.g. bacterial genomes x10^6 bp)
- Medium (e.g. lower plant genomes x10^8 bp)
- Large (e.g. mammalian and plant genomes x10^9 bp)
- Transcriptome

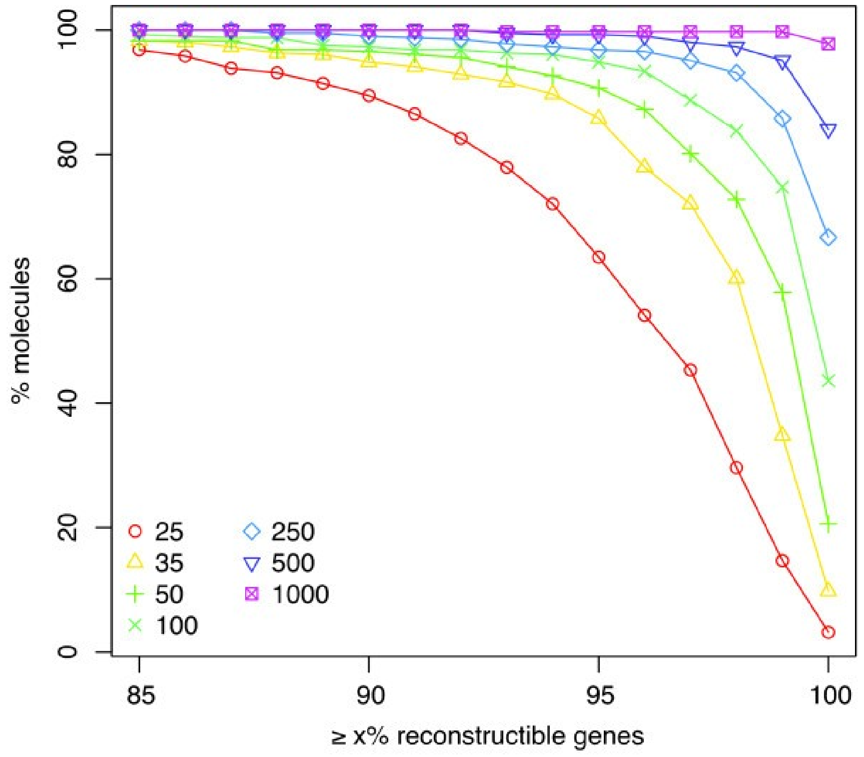
How many genomes where assembled

Kingdom	Phylum	Number of genomes
Animalia	Annelida, Arthropoda, Chordata, Tunicata, Cnidaria, Echinodermata, Mollusca, Placozoa, Porifera, Platyhelminthes, Nematoda	215
Fungi	Ascomycota, Basidiomycota, Other fungi	248
Rhizaria	Cercozoa	1
Archaeplastida	Rhodophyta	3
Chromalveolata	Cryptophyta, Heterokontophyta	18
Alveolata	Apicomplexa, Ciliophora	29
Excavata	Euglenozoa, Percolozoa, Choanoflagellatea	16
Unikonta	Amoebozoa, Metamonada	3
Plantae	Chlorophyta, Metaphyta	70

NCBI, April 2013.

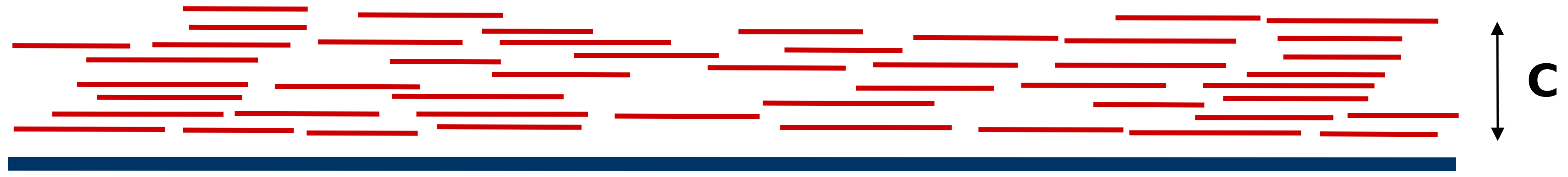
Assembling a genome draft vs complete genome

- 500 contigs covering most of a bacterial genome can be obtained in 1 week from genomic DNA to Genbank submission
- To get 1 contig covering all genomic sequence could take many months
- Is the extra effort worth it? **usually not**.



Kingsford et al. 2010

How much to sequence?



Length of genomic segment:	*L*
Number of reads:	*n*
Length of each read:	*l*

Coverage $C = n\, l \,/\, L$

How much coverage is enough?

Lander-Waterman model:

Assuming uniform distribution of reads, *C*=10 results in 1 gapped region per 1,000,000 nucleotides

Assembly approaches

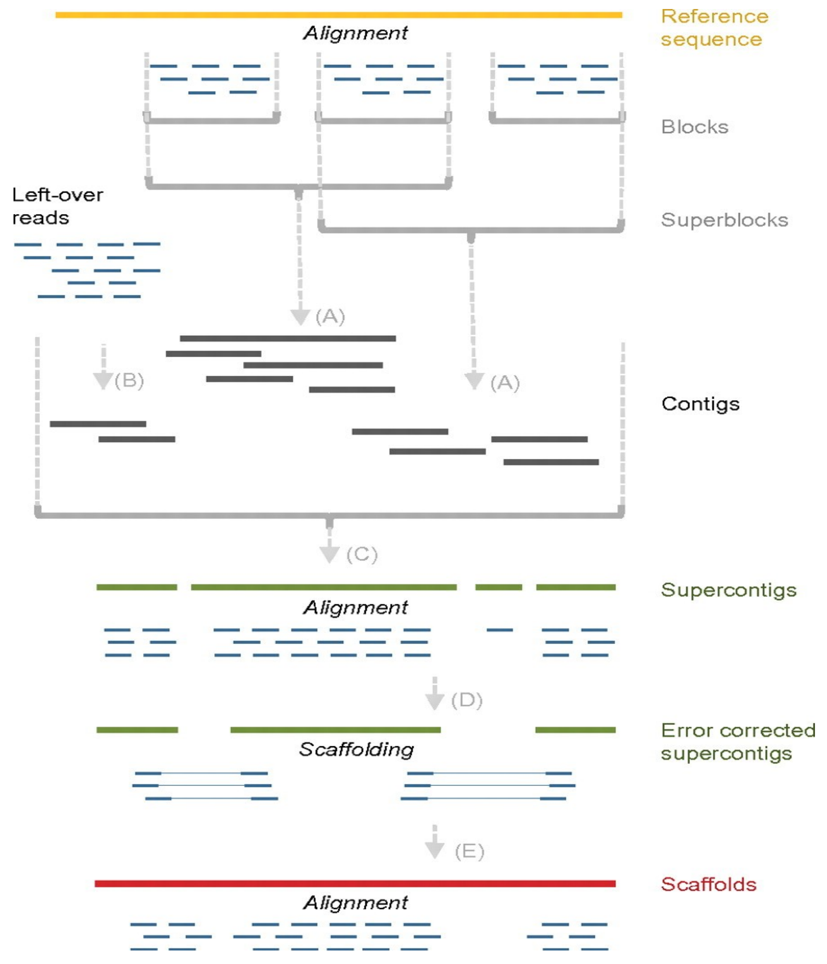
- Reference guided assembly (comparative genome assembly)

- *de-novo* Overlap Layout Consensus (OLC)
- *de-novo* De Bruijn Graph (DBG)

- Hybrid approach – *de-novo* and then reference-guided or reference-guided and de-novo for unused reads

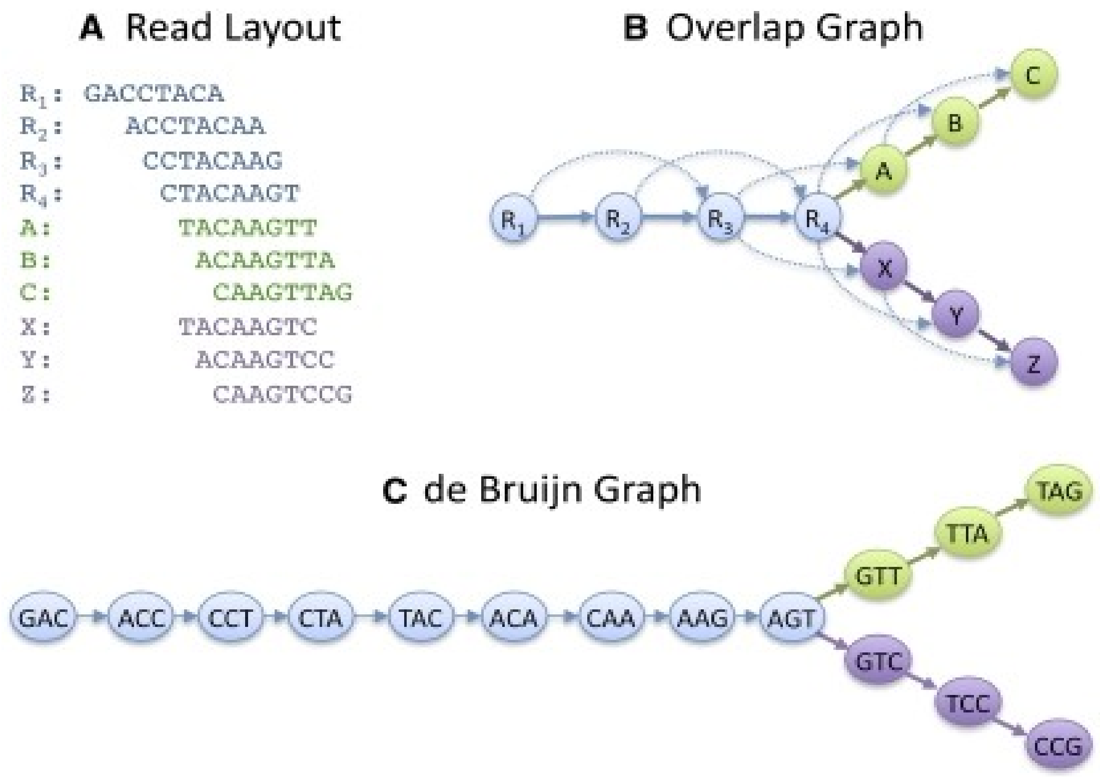
Reference guided assembly

- Divergence between assembly and reference
- Reference and data quality
- Purpose of the assembly
- Genome/Transcriptome



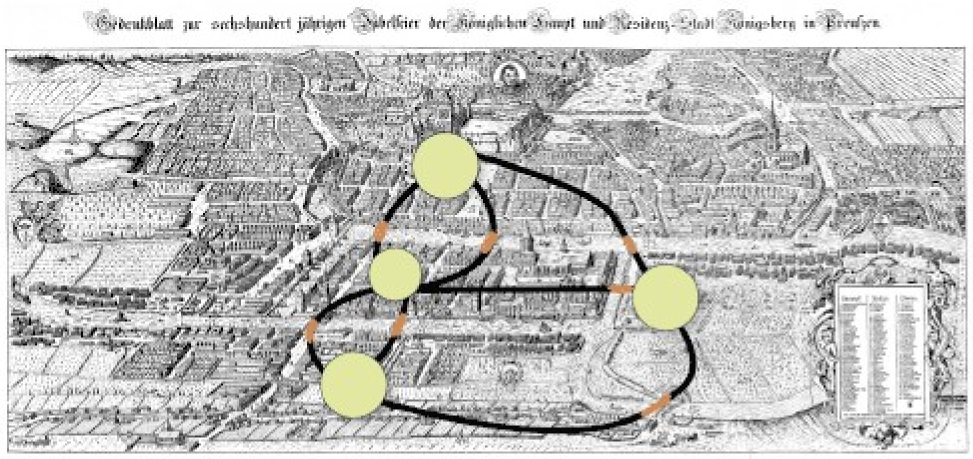
Schneeberger et al. 2011

Two paradigm for de-novo assembly



Schatz et al. 2010

Bridges of Königsberg problem



Find a tour crossing every bridge just once - *Leonhard Euler* (1735)

- A graph is *balanced* if for every vertex the number of incoming edges (bridges) equals to the number of outgoing edges: *in(v) = out(v)*

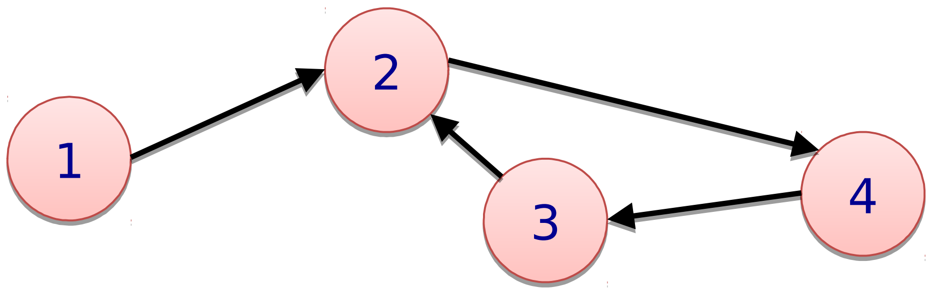
- **Theorem**: *A connected graph is Eulerian (i.e. it has an Euler cycle) if and only if each of its vertices is balanced.*

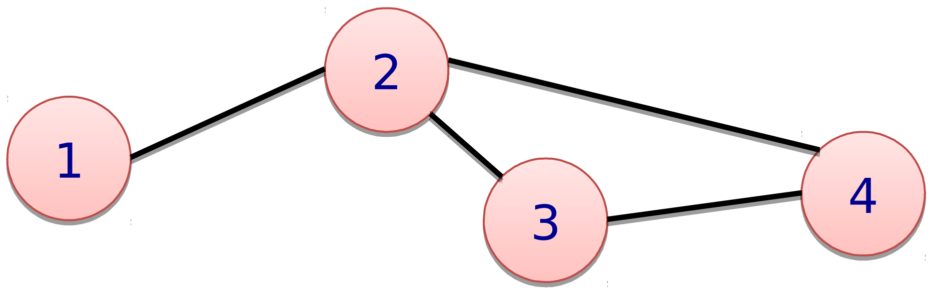
Graph theory basics

- A graph (*G*) consists of vertices (*V*) and edges (*E*)

$$G = (V,E)$$

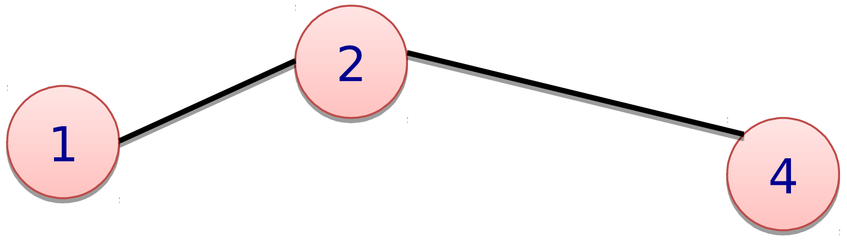
- Edges can either be *directed* or *undirected*





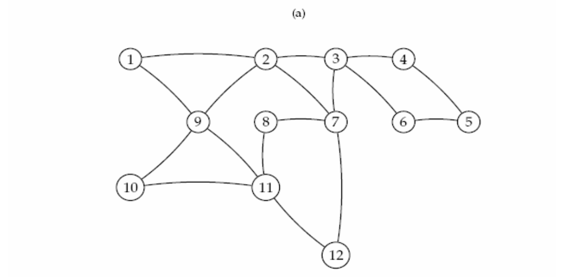
Graph theory basics

- ***Degree* of a vertex**: number of neighbours of a vertex
  - In degree: number of incoming edges
  - Out degree: number of outgoing edges
- ***path*** from u to v: number of connected edges starting from u and ending at v
- ***cycle*** is a path that begins and ends at the same vertex.

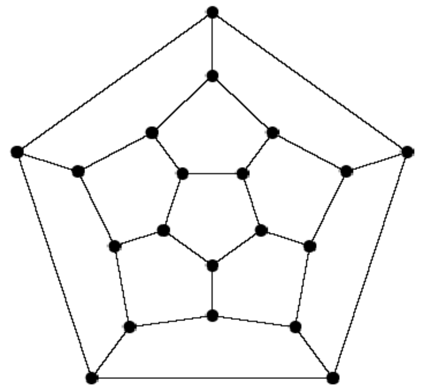


Graph types

- **Euler cycle**: find a cycle that visits every ***edge*** exactly once. Linear time



- **Hamilton cycle**: find a cycle that visits every ***vertex*** exactly once. NP-complete



What is an assembly

- Given: A set of reads (strings) $\{s_1, s_2, \ldots, s_n\}$
- Do: Determine a superstring ***s*** that “best explains” the reads

- What do we mean by “*best explains*”? **Simple/short**

Shortest Superstring Problem

- Problem: Given a set of strings, find a shortest string that contains all of them
- Input: Strings $s_1, s_2, \ldots, s_n$
- Output: A string *s* that contains all strings $s_1, s_2, \ldots, s_n$ as substrings, such that the length of *s* is minimized

- **Complexity:** NP – complete
- **Note:** this formulation does not take into account sequencing errors

Reducing SSP to TSP

- Define *overlap* (s_i, s_j) as the length of the longest prefix of s_j that matches a suffix of s_i.

```
aaaggcatcaaatctaaaggcatcaaa
                     aaaggcatcaaatctaaaggcatcaaa
```

- Construct a graph with *n* vertices representing the *n* strings $s_1, s_2, \ldots, s_n$.
- Insert edges of length *overlap* (s_i, s_j) between vertices s_i and s_j.
- Find the shortest path which visits every vertex exactly once. This is the **Traveling Salesman Problem** (TSP), which is also NP – complete.

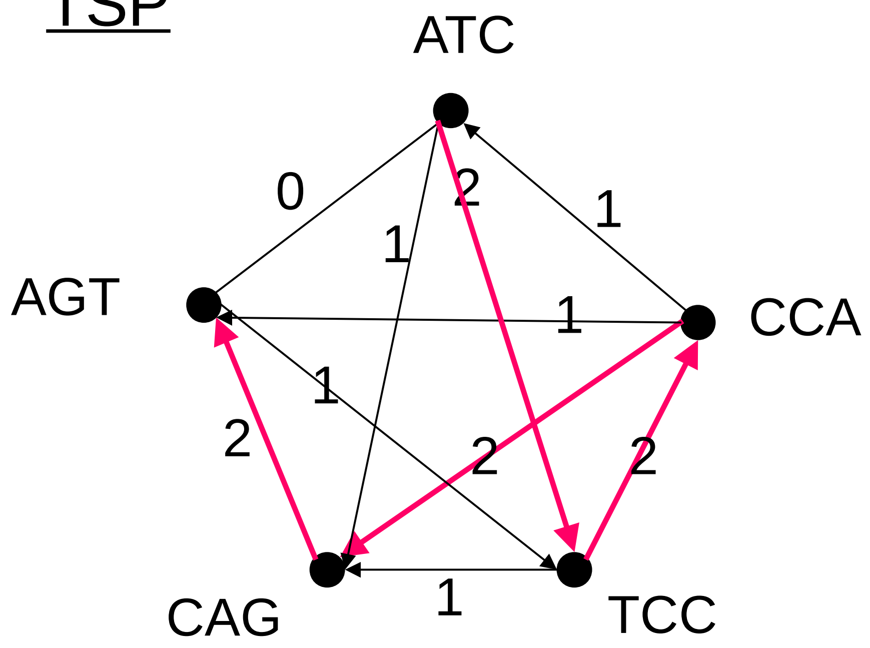
SSP to TSP: An Example

S = { ATC, CCA, CAG, TCC, AGT }

SSP

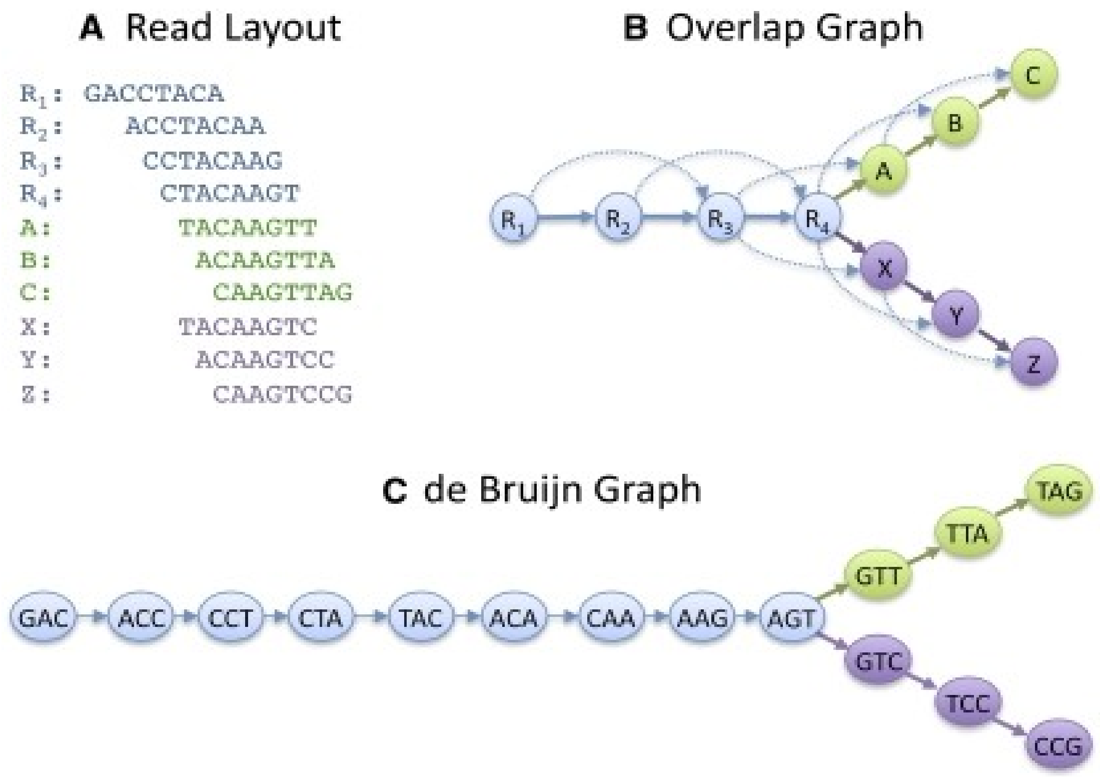
```
                AGT
            CCA
         ATC
         ATCCAGT
           TCC
              CAG
```

TSP



ATCCAGT

Two paradigm for de-novo assembly

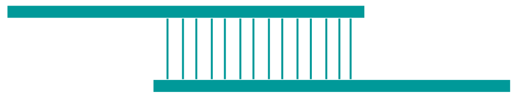


Schatz et al. 2010

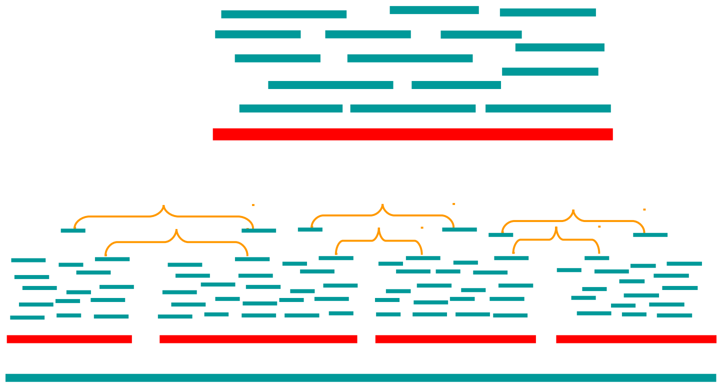
Overlap-Layout-Consensus

Assemblers: ARACHNE, PHRAP, CAP, TIGR, CELERA

Overlap (intensive): find potentially overlapping reads



Layout (simplify): merge reads into contigs and contigs into supercontigs

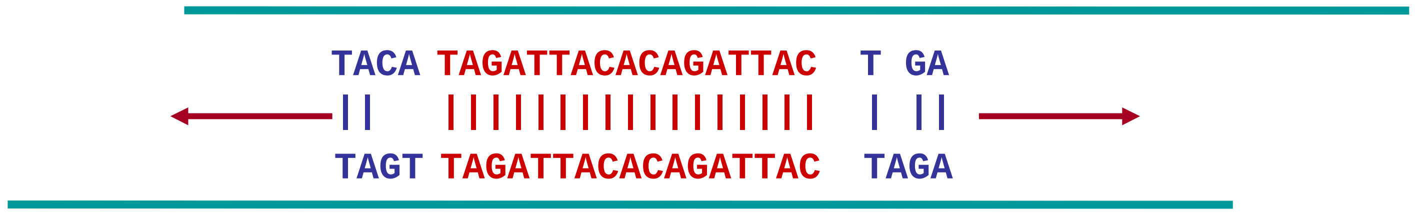


Consensus (sequence): derive the DNA sequence and correct read errors

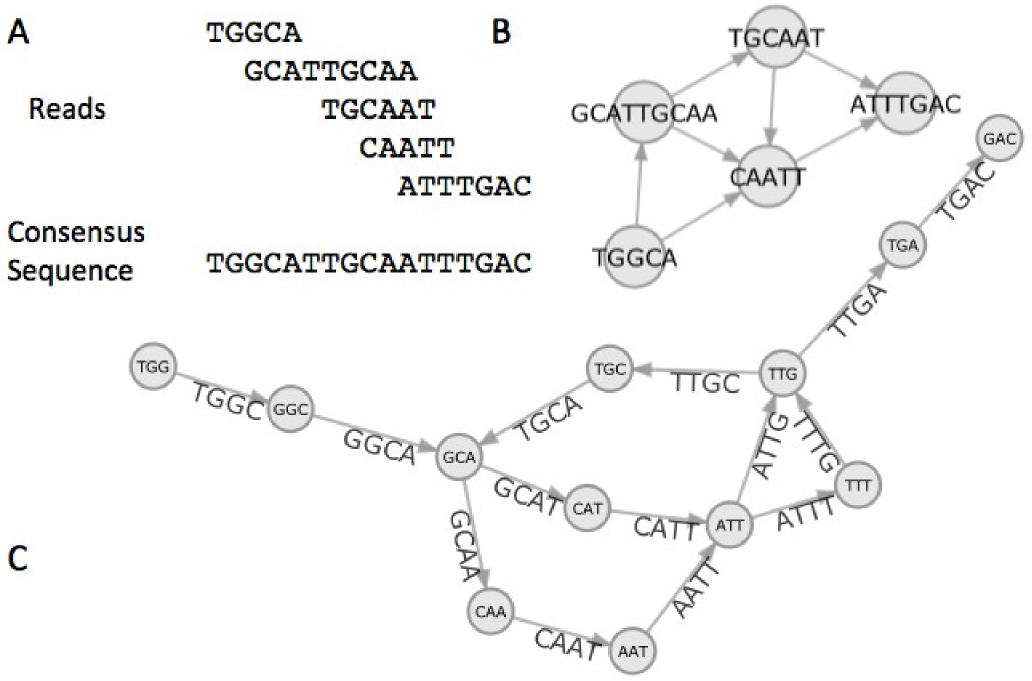
..ACGATTACAATAGGTT..

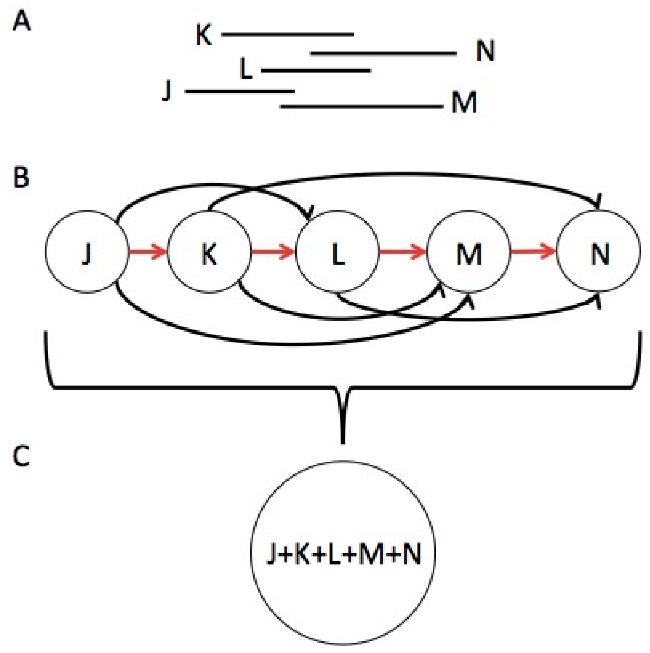
Overlapping Reads

- Find pairs of reads sharing a k-mer
- Extend to full alignment
- Set minimum threshold for similarity and overlap
- Build graph



Layout – simplify graph





Taylor 2012

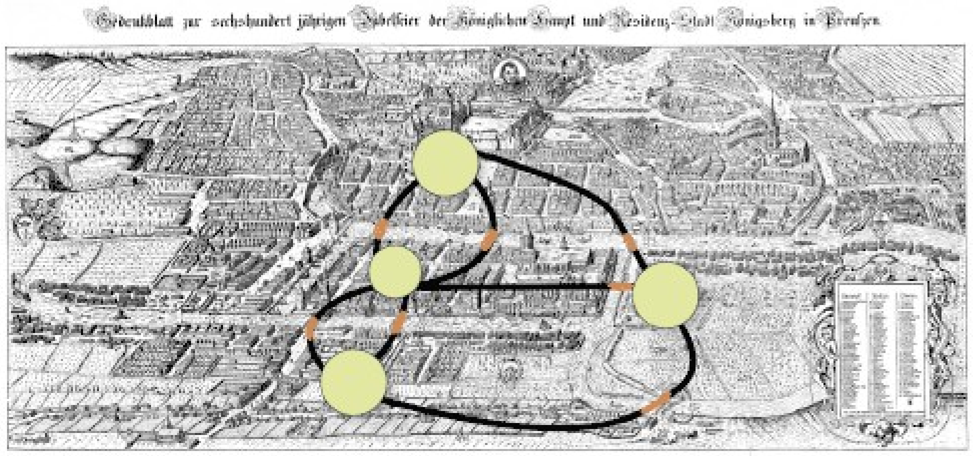
Derive Consensus Sequence

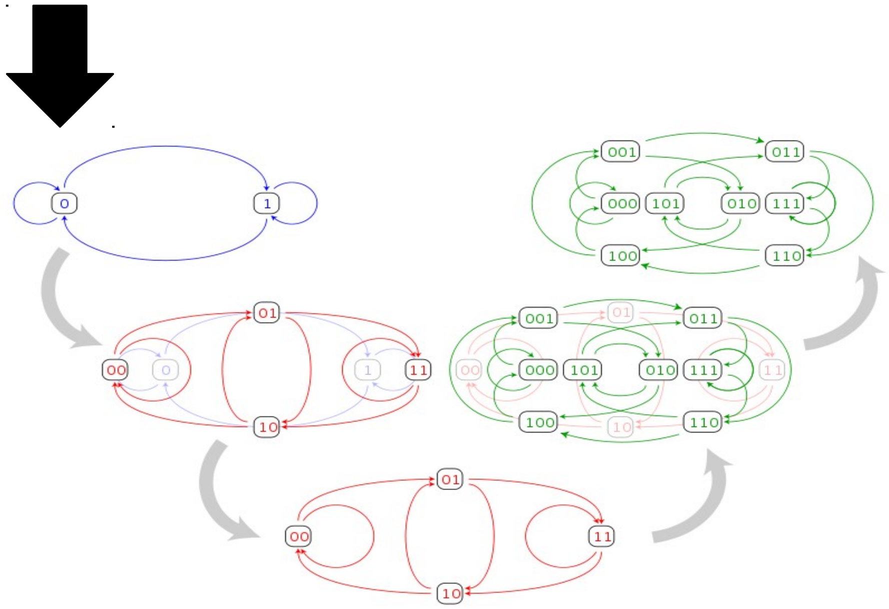
```
TAGATTACACAGATTACTGA TTGATGGCGTAA CTA
TAGATTACACAGATTACTGACTTGATGGCGTAAACTA
TAG TTACACAGATTATTGACTTCATGGCGTAA CTA
TAGATTACACAGATTACTGACTTGATGGCGTAA CTA
TAGATTACACAGATTACTGACTTGATGGGGTAA CTA

TAGATTACACAGATTACTGACTTGATGGCGTAA CTA
```

- Derive multiple alignment from pairwise read alignments
- Derive each consensus base by weighted voting

De Bruijn Graph


Gedenkblatt zur sechshundert jährigen Jubelfeier der Königlichen Haupt und Residenz-Stadt Königsberg in Preußen



The shortest circular ‘superstring’ that contains all possible ‘substrings’ of length k (k-mers) over a given alphabet

De Bruijn graph - example

“It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity,.... “

Dickens, Charles. A Tale of Two Cities. 1859. London: Chapman Hall

De Bruijn graph - example

itwasthebestoftimesitwastheworstoftimesitwastheageofwisdomitwastheageoffoolishness…

Generate random ‘reads’ of 10 bases

fincreduli geoffoolis Itwasthebe Itwasthebe geofwisdom itwastheep epochofinc timesitwas stheepocho nessitwast wastheageo theepochof stheepocho hofincredu estoftimes eoffoolish lishnessit hofbeliefi pochofincr itwasthewo twastheage toftimesit domitwasth ochofbelie eepochofbe eepochofbe astheworst chofincred theageofwi iefitwasth ssitwasthe astheepoch efitwasthe wisdomitwa ageoffooli twasthewor ochofbelie sdomitwast sitwasthea eepochofbe ffoolishne eofwisdomi hebestofti stheageoff twastheepo eworstofti stoftimesi theepochof esitwasthe heepochofi theepochof sdomitwast astheworst rstoftimes worstoftim stheepocho geoffoolis ffoolishne timesitwas lishnessit stheageoff eworstofti orstoftime fwisdomitw wastheageo heageofwis incredulit ishnessitw twastheepo wasthewors astheepoch heworstoft ofbeliefit wastheageo heepochofi pochofincr heageofwis stheageofw fincreduli astheageof wisdomitwa wastheageo astheepoch olishnessi astheepoch itwastheep twastheage wisdomitwa fbeliefitw bestoftime epochofbel theepochof sthebestof lishnessit hofbeliefi Itwasthebe ishnessitw sitwasthew ageofwisdo twastheage esitwasthe twastheage shnessitwa fincreduli fbeliefitw theepochof mesitwasth domitwasth ochofbelie heageofwis oftimesitw stheepocho bestoftime twastheage foolishnes ftimesitwa thebestoft itwastheag theepochof itwasthewo ofbeliefit bestoftime mitwasthea imesitwast timesitwas orstoftime estoftimes twasthebes stoftimesi sdomitwast wisdomitwa theworstof astheworst sitwasthew theageoffo eepochofbe

…etc. to 10’s of millions of reads

-

<u>De Bruijn solution:</u>
Represent the data as a graph (scales with genome size)

De Bruijn graph - example

Step 1:
Convert reads into “Kmers”

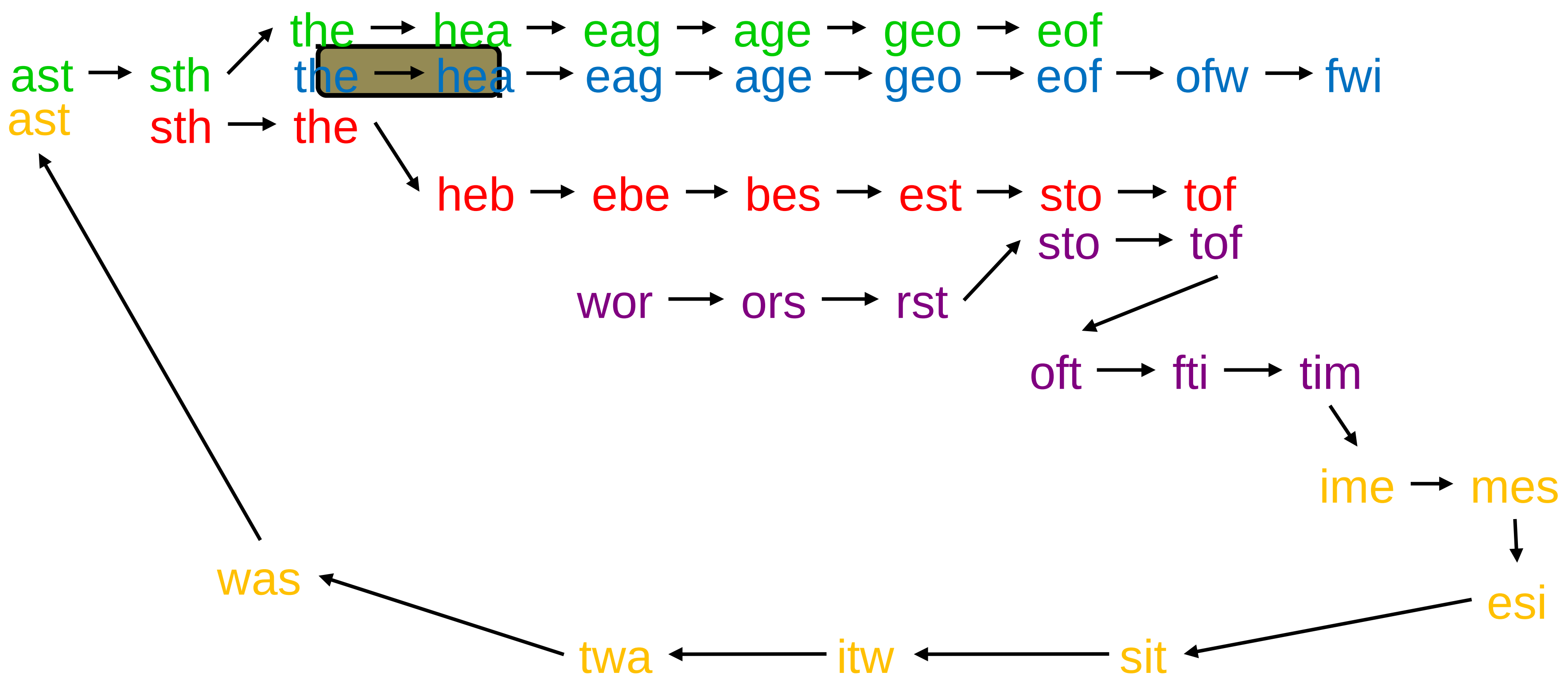
Kmer: a substring of defined length

Reads:	theageofwi	sthebestof	astheageof	worstoftim	imesitwast
Kmers :(k=3)	the	sth	ast	wor	ime
	hea	the	sth	ors	mes
	eag	heb	the	rst	esi
	age	ebe	hea	sto	sit
	geo	bes	eag	tof	itw
	eof	est	age	oft	twa
	ofw	sto	geo	fti	was
	fwi	tof	eof	tim	ast

…..etc for all reads in the dataset

De Bruijn graph - example

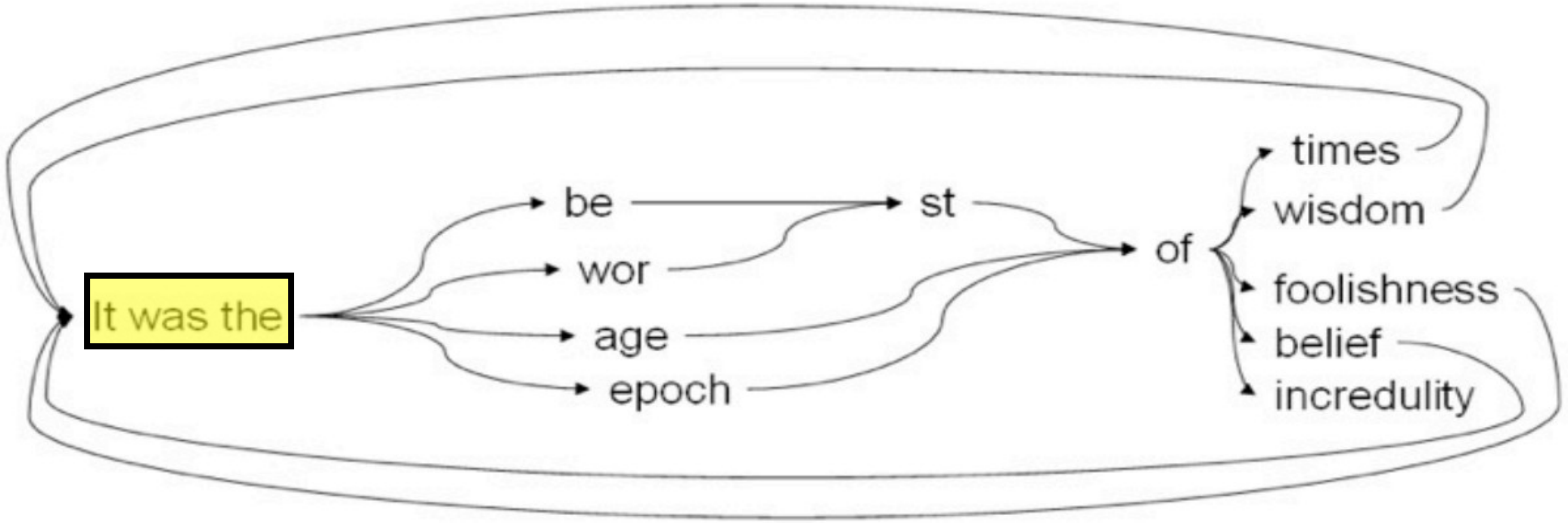
Step 2:
Build a De-Bruijn graph from the kmers (k-1 overlap)



…..etc for all ‘kmers’ in the dataset

De Bruijn graph - example

Step 3:
Simplify the graph as much as possible:

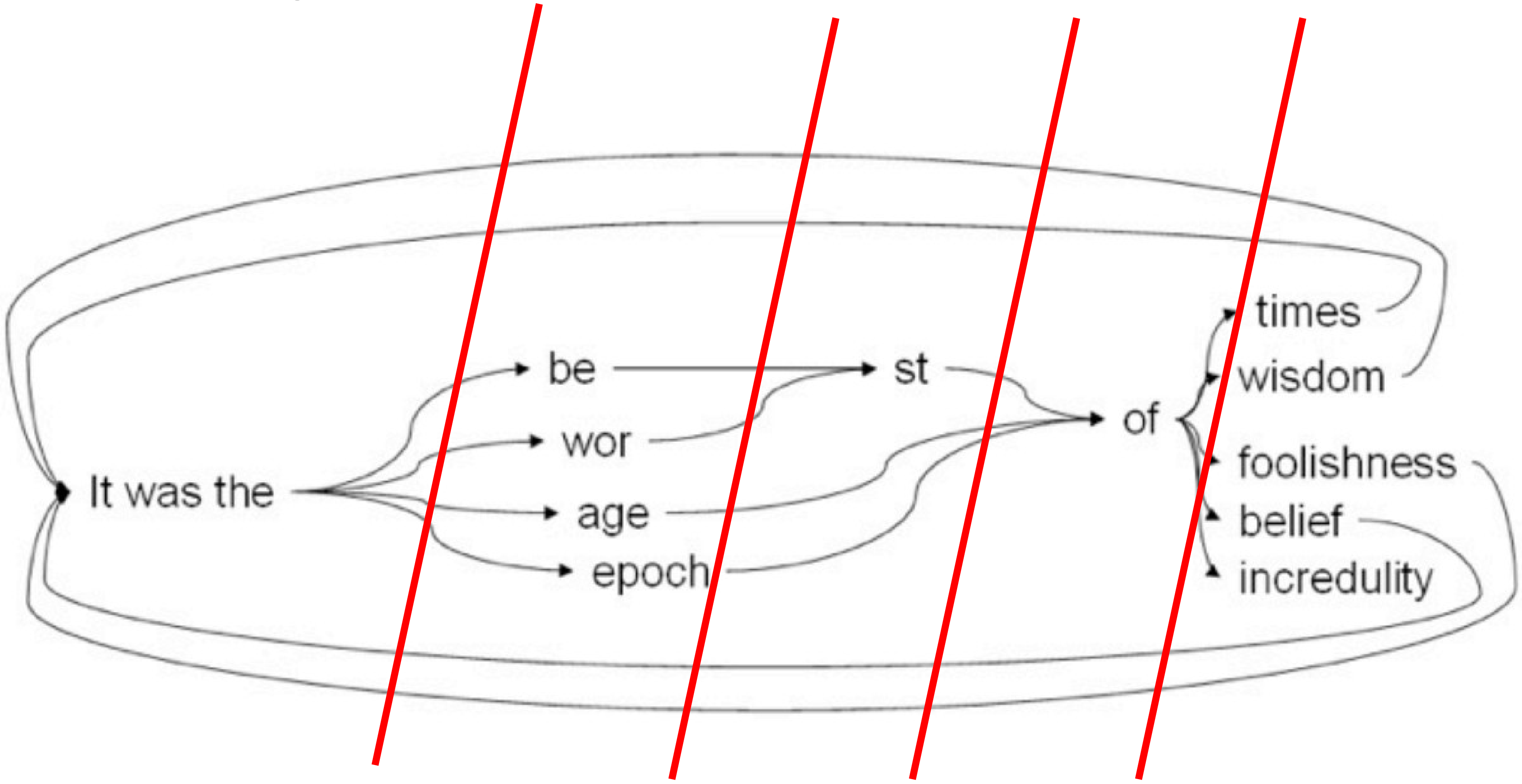


De Bruijn assemblies ‘broken’ by repeats longer than kmer

“It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity,.... “

De Bruijn graph - example

Step 4: Dump graph into consensus (fasta)



No single solution!

Break graph to produce final assembly

De Bruijn graph - example

The final assembly (k=3) - contigs

wor times itwasthe foolishness st wisdom

incredulity age epoch be of belief

Repeat with a longer “kmer” length

A better assembly (k=20)

itwasthebestoftimesitwastheworstoftimesitwastheageofwisdomitwastheageoffoolis…

Why not always use longest ‘k’ possible?

Sequencing errors:

stthebentof

$\xrightarrow{k=3}$
sth
the
heb
ebe
ben
ent
nto
tof

Mostly unaffected kmers

$\xrightarrow{k=10}$ sthebentof

100% wrong kmer

Denovo - De bruijn graph assumptions/considerations

- All k-mers present in the genome
- All k-mers are error free
- Each k-mer appears at most once in the genome
- The genome consists of a single circular chromosome

All assumptions are violated

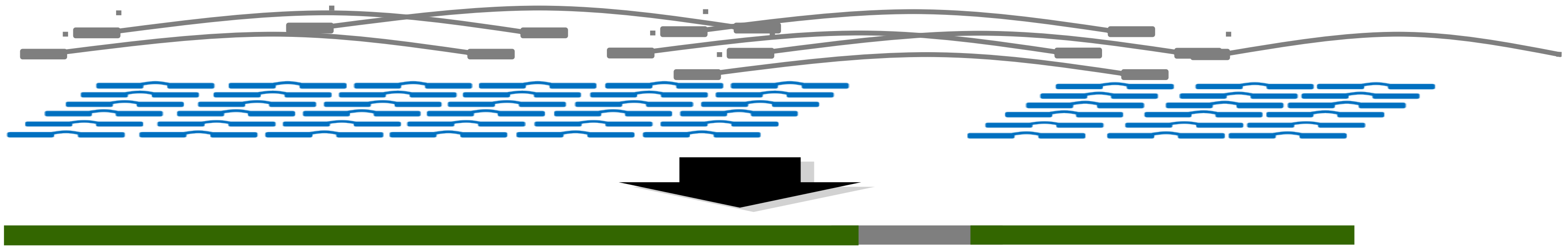
Complications for assembly

- Generating (nearly) all k-mers present in the genome
- Handling DNA repeats
- Handling multiple and linear chromosomes
- Handling unsequenced regions
- Handling sequencing errors

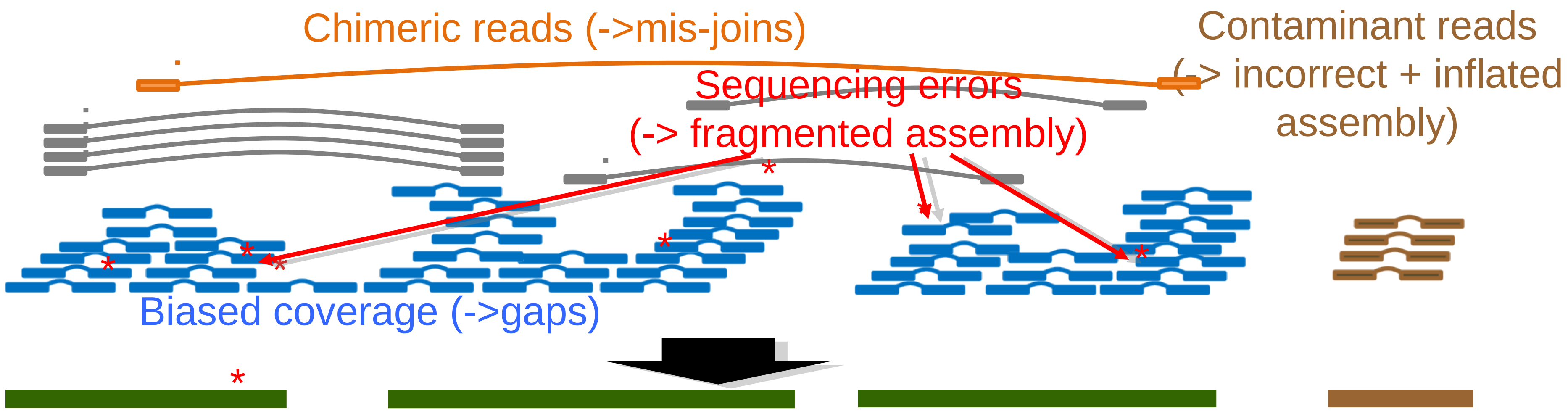
Real life assembly is messy!

<u>Assembly in theory</u>

Uniform coverage, no errors, no contamination

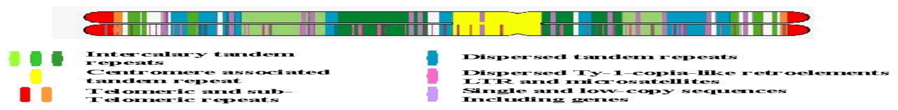


<u>Assembly in reality</u>


Chimeric reads (->mis-joins)
Contaminant reads (-> incorrect + inflated assembly)
Sequencing errors (-> fragmented assembly)
Biased coverage (->gaps)

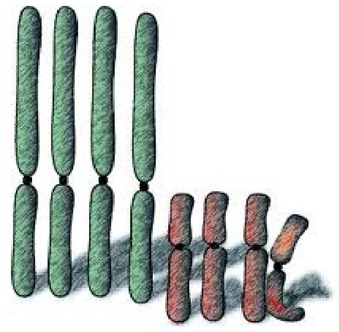
Real life assembly is messy!

High repeat content


Intercalary tandem repeats
Centromere associated tandem repeat
Telomeric and sub-Telomeric repeats
Dispersed tandem repeats
Dispersed Ty-1-copia-like retroelements
LTR and microsatellites
Single and low-copy sequences including genes

**RESULT:
misassemblies / collapsed assemblies**

Polyploidy



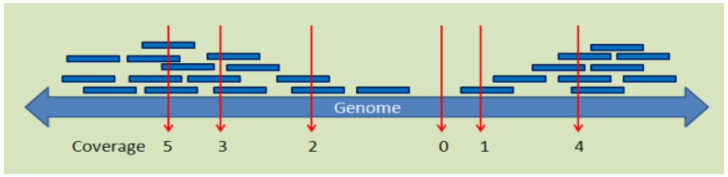
**RESULT:
fragmented assembly**

Biased sequence composition

ACTGTCTAGTCAGCGCGC
GCGCGCGCGCCCGCGCG
CGCGGGCGGCGGCGCGG
GCGGGCGCATGTAGTGAT

**RESULT:
incomplete / fragmented assembly**

Non-uniform coverage



**RESULT:
Incomplete / fragmented assembly**

How to solve complications

velvet (Zerbino & Birney 2008):

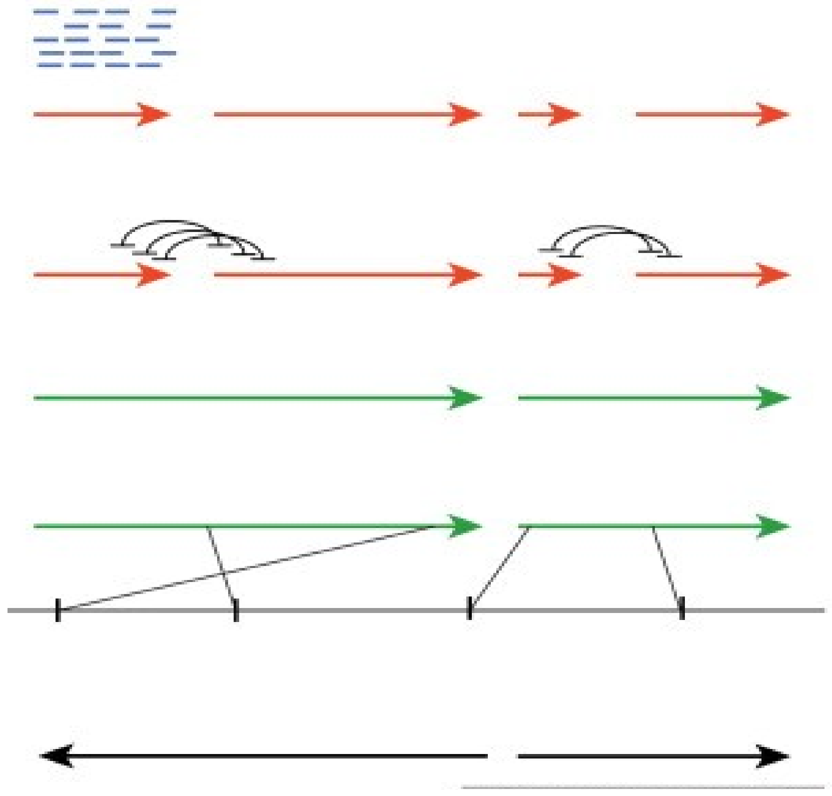
- Build graph
- Remove errors:
  - “tips”
  - “bubbles” - Tour Bus
  - Erroneous connection
- Solve repeats: breadcrumbs

AllPaths-LG (Gnerre et al. 2011):

- Correct errors
- Fragment pair filling
- Build graph
- Gap patching
- Flattening
- Scaffolding

Scaffolding

- Scaffolding represents the task of ordering and orienting contigs by using additional information about their relative placement
- Mate-reads information, homology data, physical maps, or gene synteny



Ellegren 2014

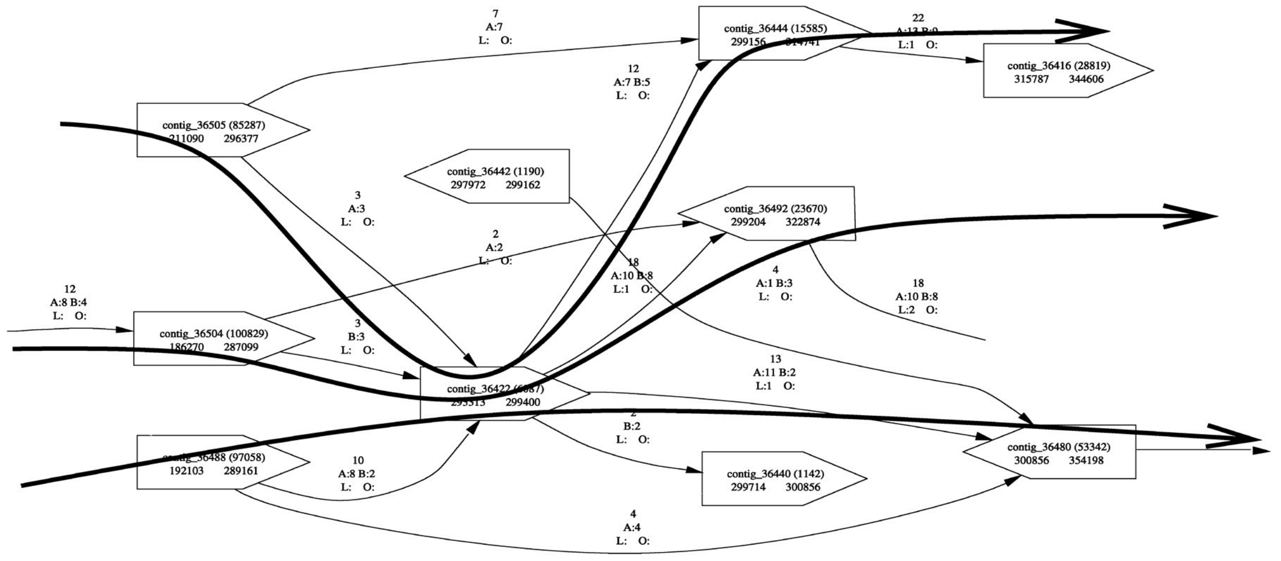
Mate-pair vs paired-end

- Paired-end usually refers to libraries prepared for the Illumina platform with insert sizes 50-500bp and forward-reverse orientation (→ ←).

- Mate-pair is a different library preparation protocol and usually produces insert sizes 2kb-20kb and reverse-forward orientation (← →).

Scaffolding algorithm

- Find links
- Filter links: insert size, minimum support threshold
- Set orientation: majority rule
- Traverse graph


contig_36444 (15585) 299156 314741
contig_36416 (28819) 315787 344606
contig_36505 (85287) 211090 296377
contig_36442 (1190) 297972 299162
contig_36492 (23670) 299204 322874
contig_36504 (100829) 186270 287099
contig_36422 (6087) 293313 299400
contig_36488 (97058) 192103 289161
contig_36440 (1142) 299714 300856
contig_36480 (53342) 300856 354198

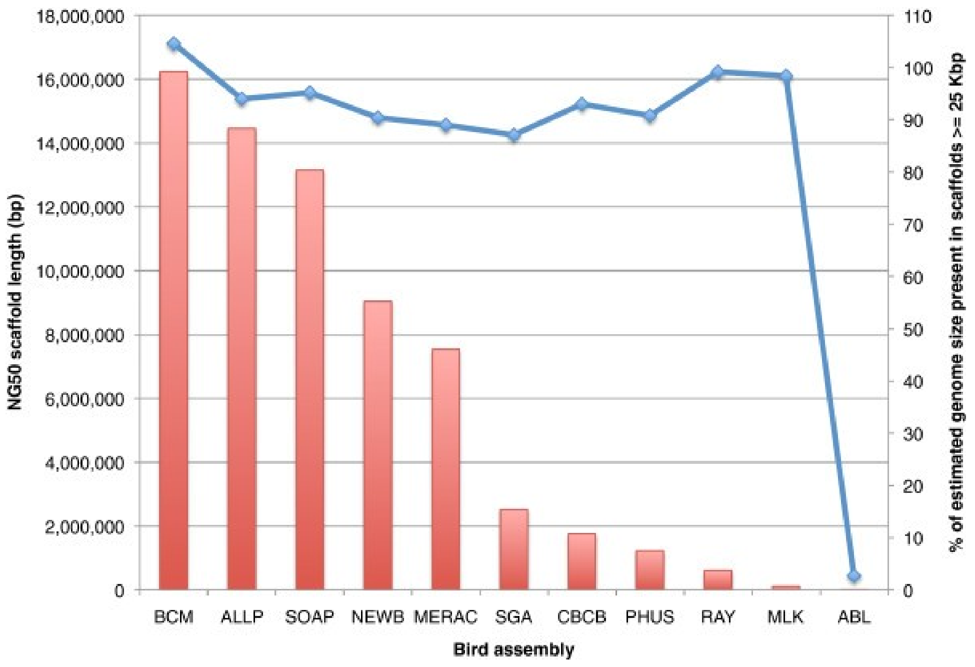
Pop et al. 2004

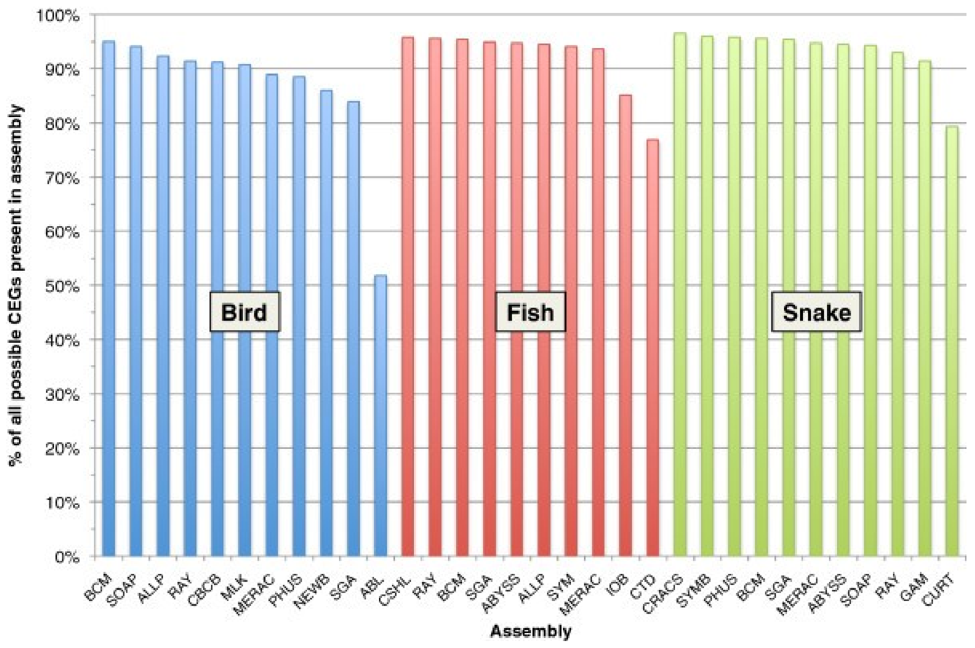
Evaluation

- Assembly length
- N50: length of a contig/scaffold (N) for which 50% of all bases in the assembly are in a sequence of length L < N
- Number of contigs/scaffolds
- Longest contig/scaffold
- Proportion of gaps (N's)
- Re-mapping (ALE, REAPR, Hagfish...)
- Genes

Comparison

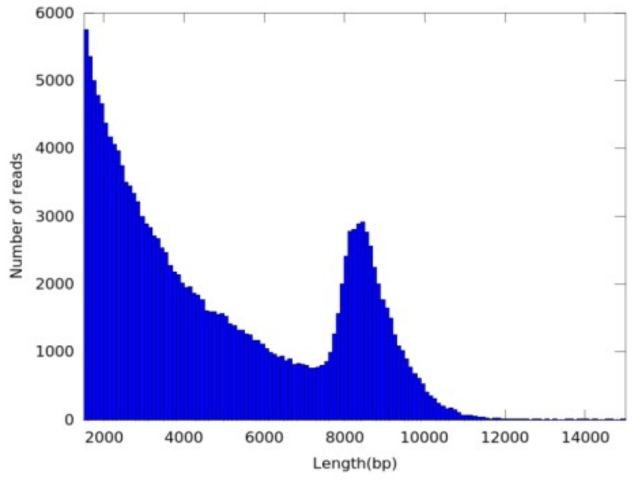
The Assemblathon2

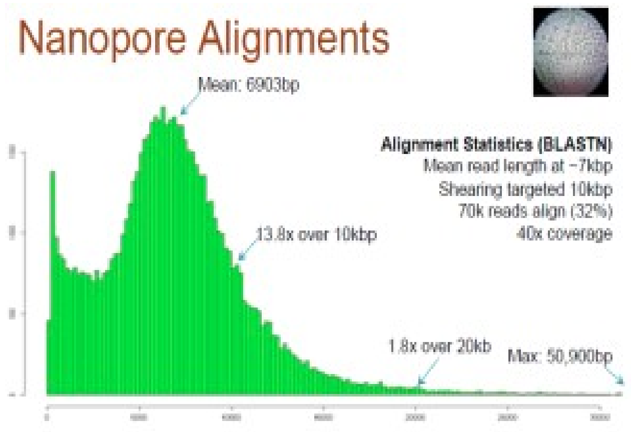




Bradnam et al. 2013

Other technologies (long reads)





Alignment Statistics (BLASTN)
Mean read length at ~7kbp
Shearing targeted 10kbp
70k reads align (32%)
40x coverage

PacBio read length distribution

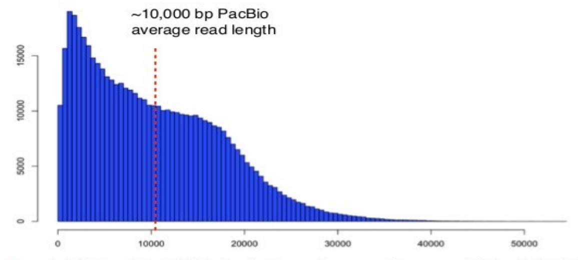
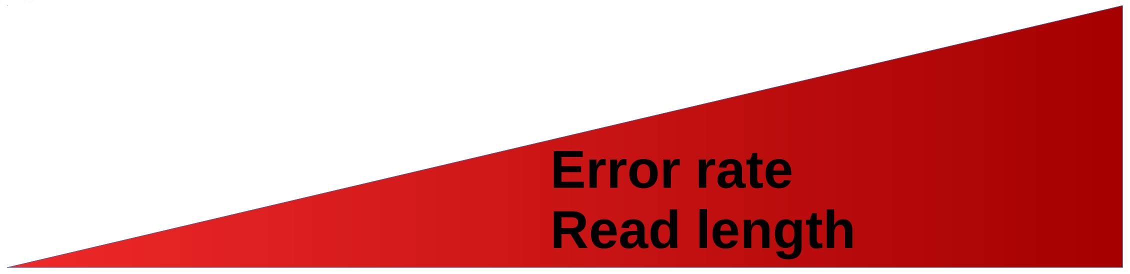


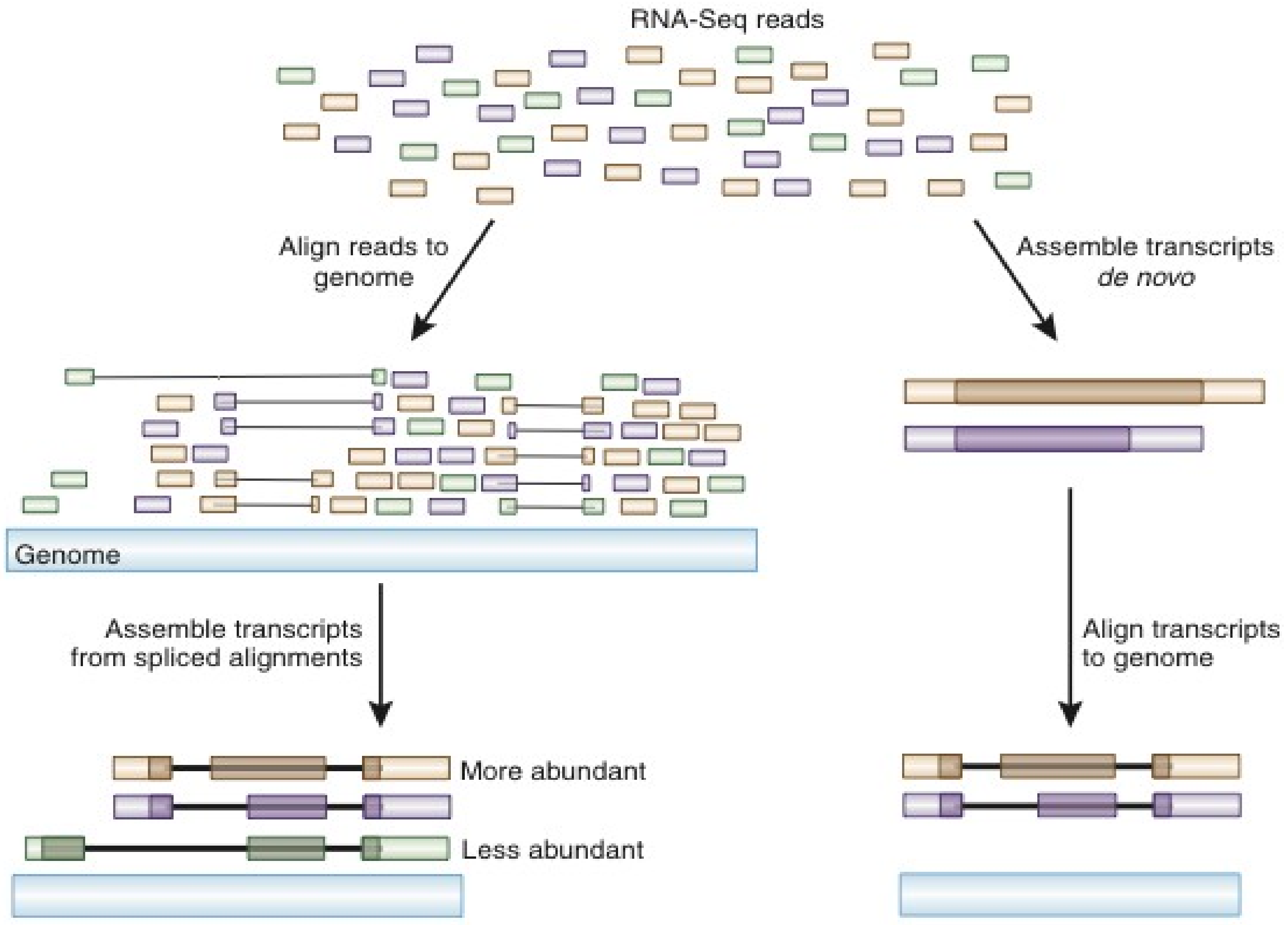
Figure 1: PacBio read length distribution for the recently sequenced rice genome (IR64) using C3-P5 chemistry sequenced at PacBio. The mean read length was 10,232 bp, and the maximum extends to 54,288bp.



Transcriptome assembly

- Cheap (1-2 lanes for most applications)
- Fast
- Informative
- Expression profiling

There is no one correct solution



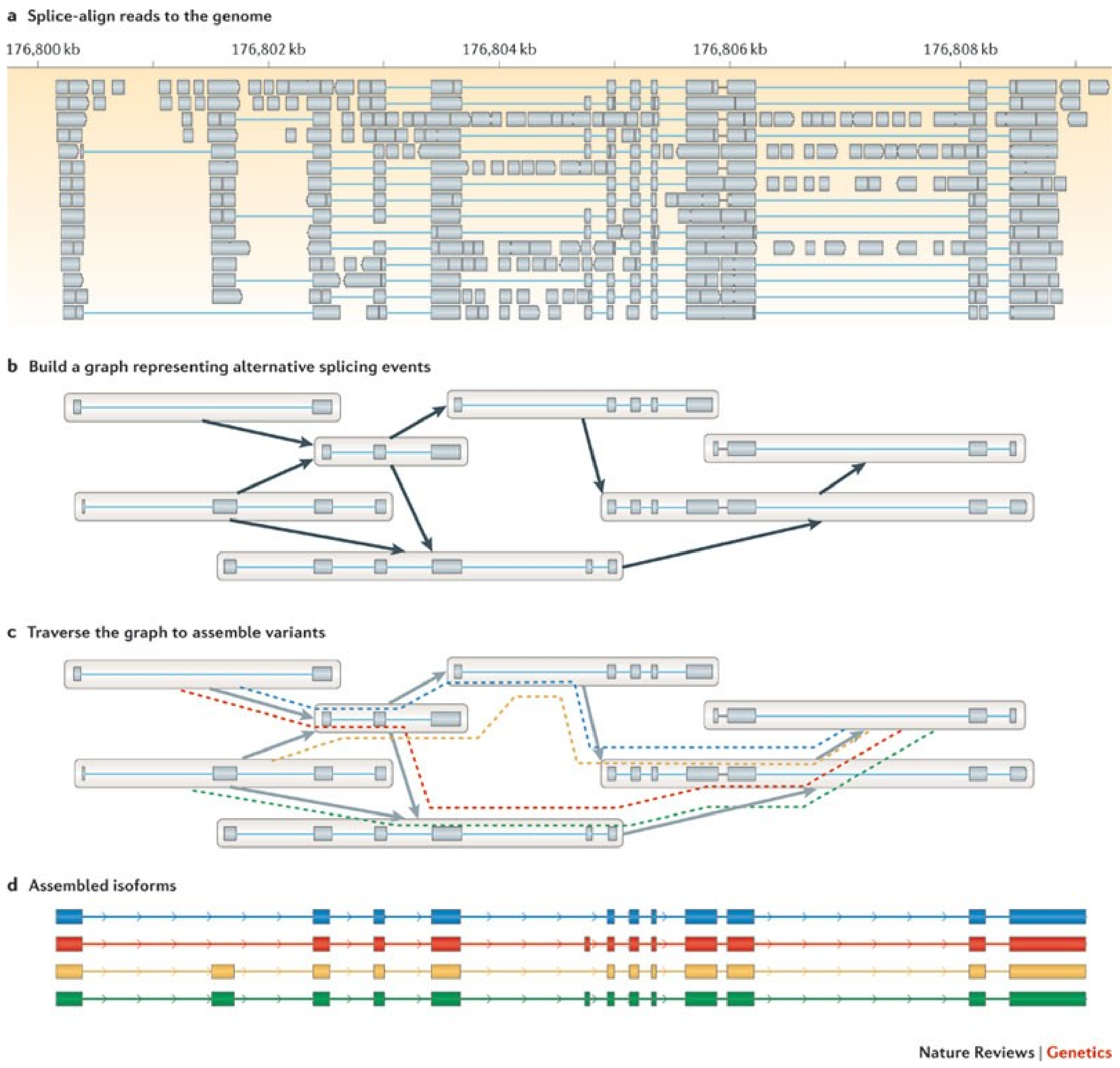
RNA-Seq reads
Align reads to genome
Assemble transcripts *de novo*
Genome
Assemble transcripts from spliced alignments
Align transcripts to genome
More abundant
Less abundant

Haas & Zody 2010

Challenges for transcriptome assembly

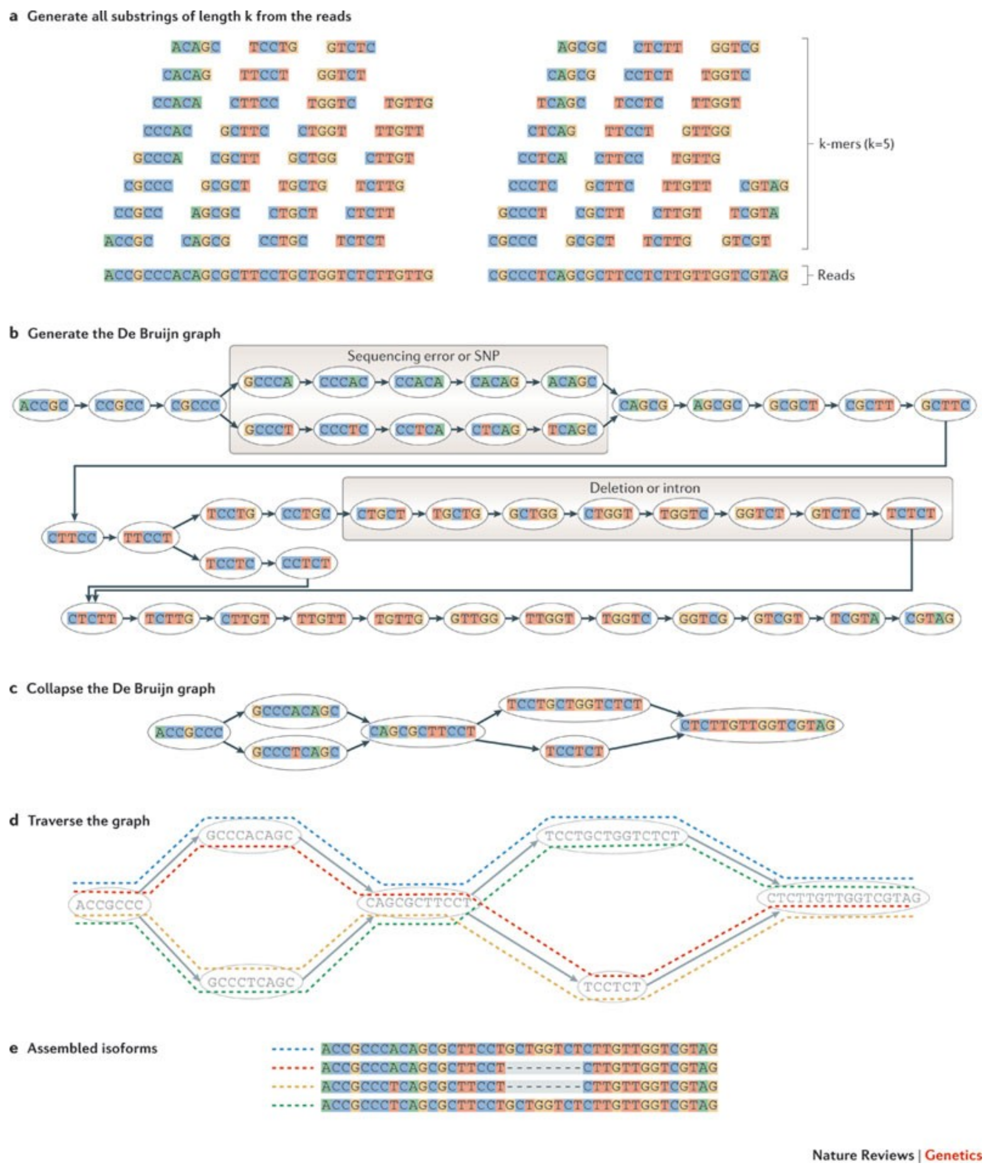
- Highly non-uniform coverage
- Alternative splicing
- Alternative promoter usage
- Alternative poly(A)

Reference-guided transcriptome assembly


a Splice-align reads to the genome
176,800 kb
176,802 kb
176,804 kb
176,806 kb
176,808 kb
b Build a graph representing alternative splicing events
c Traverse the graph to assemble variants
d Assembled isoforms
Nature Reviews | Genetics

Martin & Wang 2011

De novo transcriptome assembly



Martin & Wang 2011

Assembly cookbook

A genome assembly project, whatever its size, can generally be divided into stages:

0) Why? What would be considered as a success?

1) Experiment design

2) Sample collection

3) Sample preparation

4) Sequencing

5) Pre-processing

6) Assembly

7) Post-assembly analysis

Assembly cookbook

1) Experiment design:

- What is known about the genome?
- How big is it?
- How repetitive?
- Polyplidy?
- Heterozygosity?
- Other sources are available: close relatives, maps, databases, etc.
- Sample: single cell, pool, meta-genomes
- Computation: memory, CPUs, software, collaborations
- Budget

Assembly cookbook

5) Pre-processing::

- Quality trimming
- Adapter clipping
- Error correction (QUAKE, ECHO, Illumina reads)
- Merge overlapping paired-end (COPE, FLASH)
- Removal of other undesirable sequences: contaminations

Assembly cookbook

6) Assembly::

- Assemble few versions (software, K-mer, parameters)
- Think well and consult before setting parameters
- Small/medium genomes: PacBio have a protocol to close small genomes, Mira, A5, and many more...
- Big genomes: if long reads: OLC, if short reads: if heterozygote: Platanus/SOAPdenovo2, if homozygote: AllPaths (data specifications) or SOAPdenovo2, if mixture of short and long: Mira (OLC), Ray (DBG).

Assembly cookbook

7) Post-assembly:

- Compare assemblies: N50, max, N's, total length
- Re-map reads: coverage, if possible use ALE, REAPR, etc.
- Map transcripts/known/conserved genes to the assembly
- Merge assemblies if possible
- Annotation
- Compare to other related species

Exercise

- Use the same data as yesterday:
- Assemble the 1% error rate library with Kmer-31/21
- Total length
- N50
- N90
- Number of contigs
- Map reads to reference – how many map/properly