Genome bioinformatics 1

What bioinformaticians do today…

November 8, 2016

Outline

- Intro: history, sequencing technologies, main concepts (molecular biology, bioinformatics)
- Whole genome sequencing tasks
  - Genome assembly: Short read strategies, De Bruijn graphs

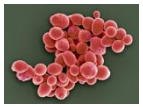
1 INTRODUCTION

INTRO

Quick reminder, you can skip it

Timeline of discovery

1st public discussion of Human genome sequencing project



Yeast genome sequenced

C. elegans (roundworm)

Mouse

Several ten thousand organisms sequenced!

1984

1995

1996

1997

1998

2000

2002

2016

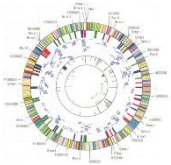
1st bacterial genome sequenced

HTG Human genome sequencing begins

E. coli

Human, fly genome drafts

Sequencing enters healthcare, whole patient genomes

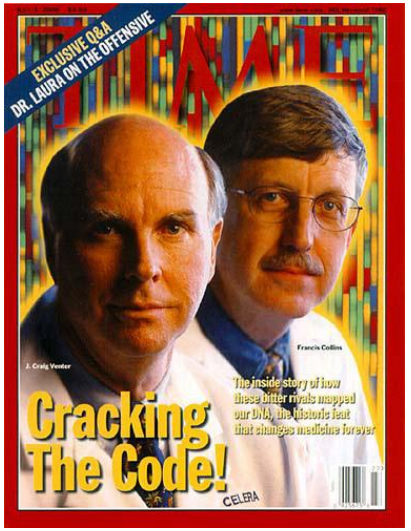


Quick reminder, you can skip it

The politics of the Human Genome project

- Public Project (NIH):
  - Mapped long chunks of the genome first (to sort out where the repeats were), then shotgun sequenced these
  - Started 1990, finished 2003
  - Cost: ~$3 billion

- Business Project (Celera):
  - Shotgun sequenced *entire* human genome in one go
  - Started in ~1998
  - Cost: ~$300 million
  - Used the public project data to help assembly

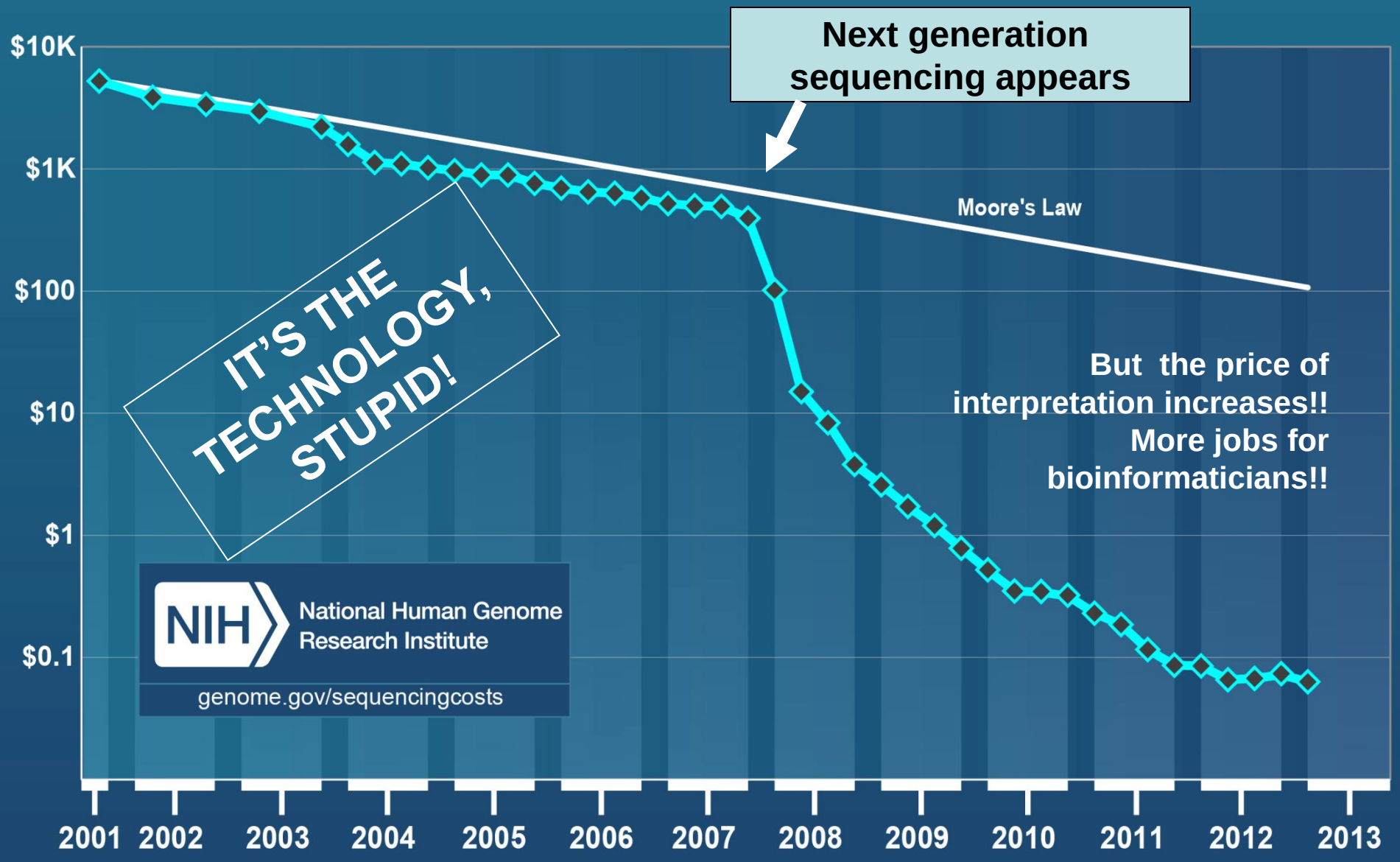


Craig Venter (Celera)

Francis Collins (public)

INTRO

Cost per Raw Megabase of DNA Sequence



INTRO

Why should we sequence genomes of various organisms?

- To catalog all the genes present in one organism.
- To compare the gene content of one organism to another organism.
- To study features other than genes.
- To study genome evolution.
- To study organismal evolution.
- As a foundation for future experimentation.

To know nature better

INTRO

Why should we know the human genome?

- To catalog all the genes present in in human .
- To find mutations, variations, disease related genes.
- To understand an cure genetic diseases.

To improve human health

The majority of non-human sequencing projects is also related to human health: animal models

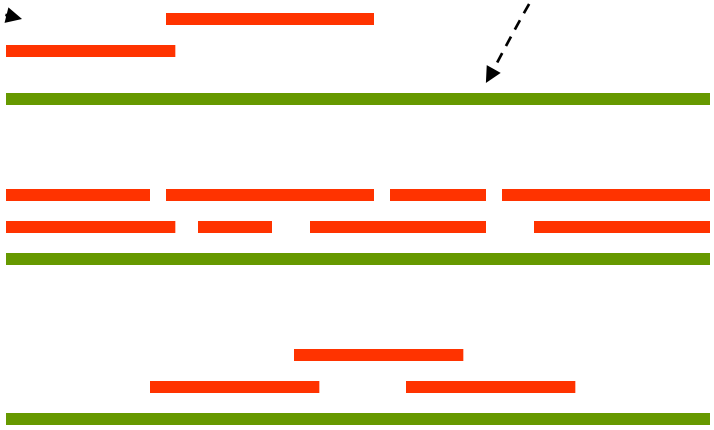
Genome sequencing problems:
Two fundamental tasks

- Assembling short sequence pieces (“reads”) into larger pieces, incl. genomes

- Finding mutations, by mapping reads to known genomes

Vocabulary

- Read: A single piece of output by a sequencer (typically a 50-500bp long DNA sequence).
- Coverage: The number of times a (genome) sequence is covered with reads. Sequence coverage is the fraction of the genome covered by reads.



Coverage ~ 0.5

Coverage ~ 2

Sequence coverage ~ 0.5

For difficult problems (disease mutations) we need very high coverage (up tohundreds e.g.)

Vocabulary

- Fragment library: a library of reads with short (<1000 nucleotide) „insert” sizes. Also known as *std* library
- Long insert library: A library of reads with long (4-8kb) insert size where only 100 bp on each end are sequenced. Also known as CLIP or mate pair library. Contains unsequenced parts in the middle!
- Contig: A contiguous sequence of DNA (assembled from single reads)
- Scaffold: One or more contigs linked together by unknown sequence segments
- Captured gap: A gap within a scaffold. The order and orientation of the contigs spanning the gap is known

GENOME

Parts of a genome sequencing project

- DNA sequencing
- Assembly*
- Gene prediction/gene annotation*
- Data deposition*

*bioinformatics tasks.

Note: A similar genome is a big help. In health applications we seaquence only parts of a known genome (human).

Sequencing projects: prokaryotes vs. eukaryotes

- Prokaryotes (e.g. bacteria, viruses) are simple to sequence: they contain many genes, few repeats, little intergenic parts
- Even a mixture of prokaryotes can be sequenced for species identification purposes (this is called metagenomics)
- But: all genomes are >50% different
- Eukaryotes (including humans) are difficult, many repeats, exons, introns etc.

INTRO

Sequencing technologies

"Traditional sequencing technologies"

Technology	Run time	Read Length	Throughput	Error
Sanger Gel	6–8 hours	700bp	67.2 kb/hr	0.0001%
Sanger Capillary	1–3 hours	700bp	166 kb/hr	0.0001%
454/Roche FLX	4 hours	200-300bp	20-30 Mb/hr	0.80%
Illumina/Solexa	2–3 days	30-100bp	20 Mb/hr	0.10%
ABI/SOLiD	8 days	35bp	5-15 Mb/hr	0.06
Ion Torrent	2 hours	~200bp	5Mb/hr	2%
Pacific Biosciences	2 hours	2000-4000bp	0.5Mb/hr	10-12%

"Next generation sequencing"

base, not byte

Sequencing “platforms”

- A sequencing method includes not only the sequencer (i.e. the apparatus), but also a specific workflow (sample preparation, data analysis, etc). Altogether this is referred to as “platform”.
- What is important for us: how long are the reads and what is the accuracy of the reading (error%). (The rest is not bioinformatics).

INTRO

Summary: traditional vs. next generation sequencing (NGS)

- **Traditional (Sanger) sequencing**
  - Accurate
  - Also works on few samples
  - Expensive for data
  - Small capital investment
  - Slow
- **Next generation sequencing**
  - Less accurate – sometimes much less.
  - Shorter reads (in general)
  - Economical only with many samples
  - ~1000 less expensive for data
  - Large capital investment
  - Very fast

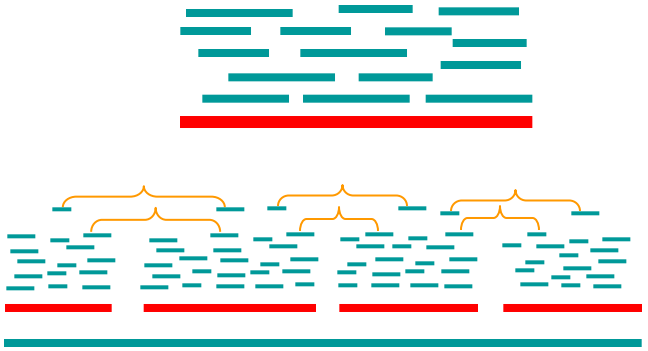
Today, large labs typically work on very high throughput NGS machines (Illumina), smaller labs use medium throughput, versatile machines (Ion Torrent).

INTRO

Sequence assembly

Overlap: find potentially overlapping reads

Layout: merge reads into contigs and contigs into supercontigs



Consensus: derive the DNA sequence and correct read errors

..ACGATTACAATAGGTT..

INTRO

The mathematical problem

- We start with thousands of DNA reads, 200 bases each
- Multiple copies of DNA provide multiple coverage by reads
- The problem of genome assembly is to recover the original sequence of bases of the genome (as much as possible…). There is generally no other information available.

2 WHOLE GENOME SEQUENCING (WGS)

WGS problem1 De Novo (there is no similar genome)

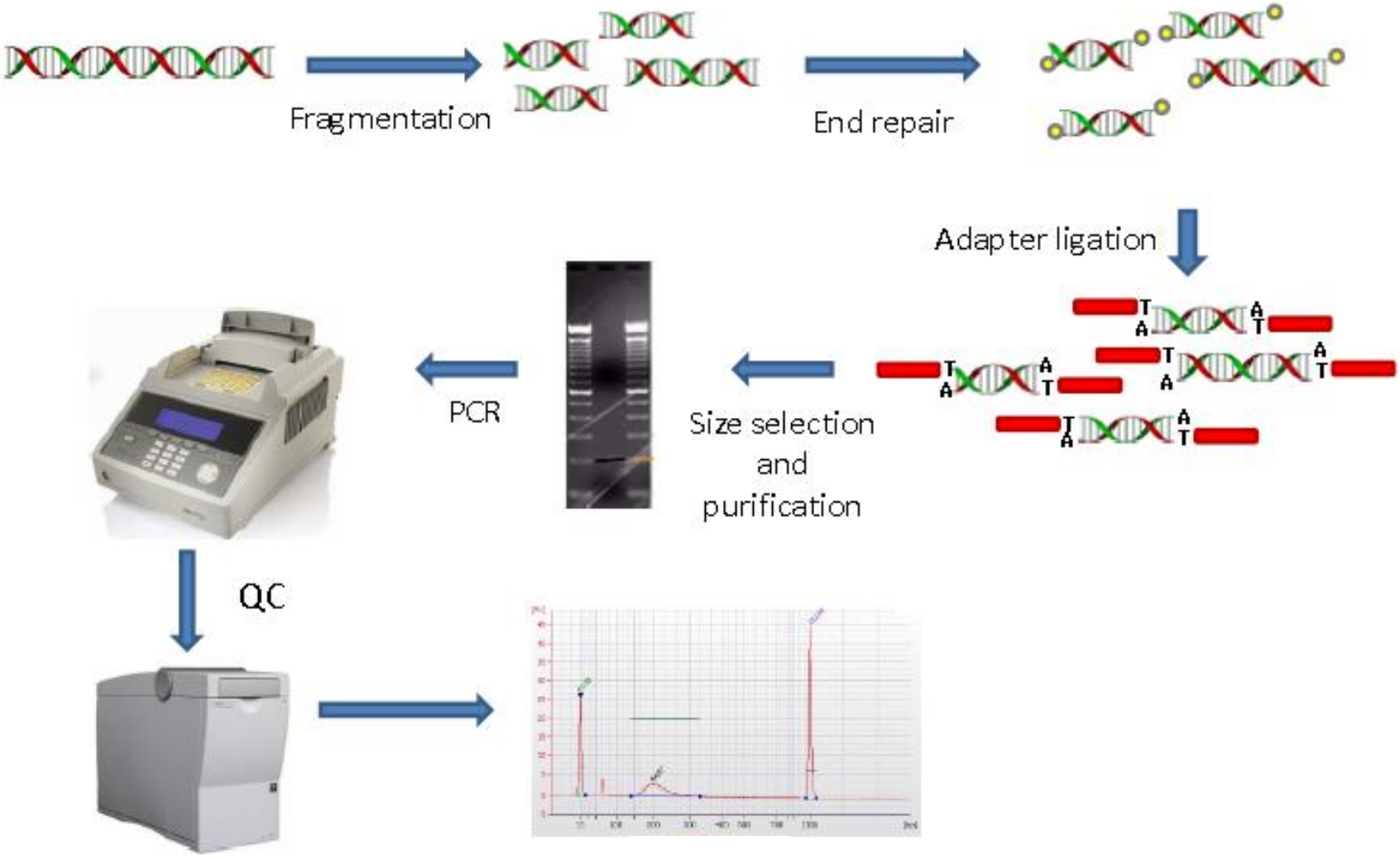
WGS problem 2 Using a similar, so-called reference genome

GENOME

DNA LIBRARY CONSTRUCTION

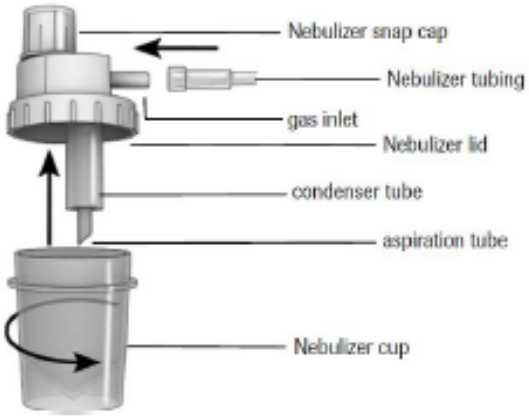
GENOME

DNA library preparation



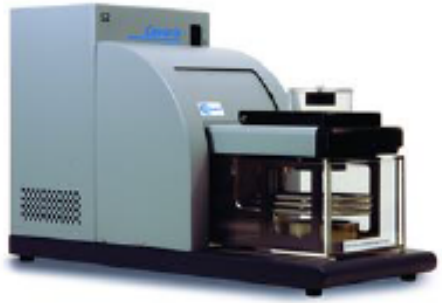
GENOME

DNA fragmentation



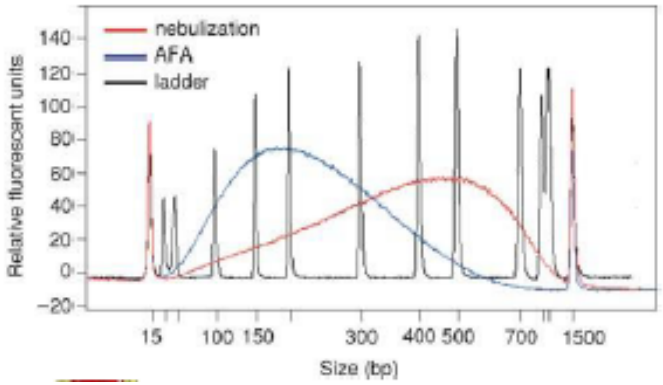
Nebulization:

- Cheap
- Reproducible results
- Produces a wide size distribution of fragments



Covaris S2:

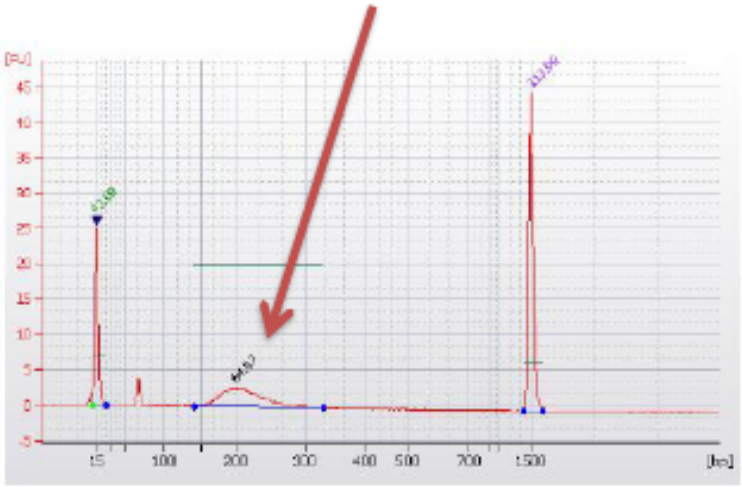
- Much narrower size distribution
- Expensive
- Is included in SOLiD 4 system (for ePCR)

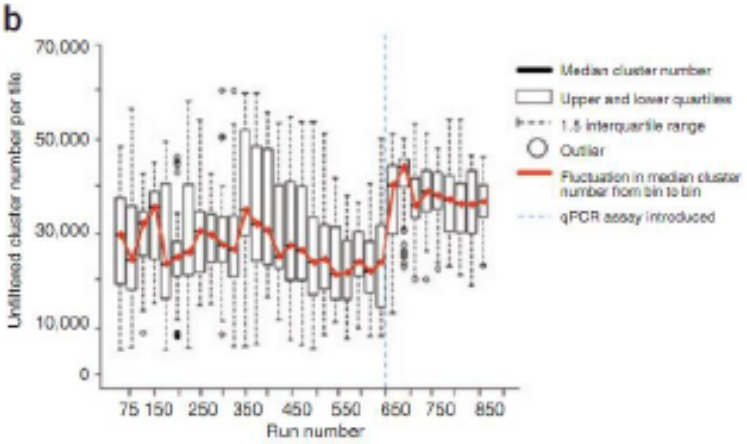


GENOME

Library quality control and quantification

Library size and distribution





Library quality is evaluated by Bioanalyzer analysis with DNA High Sensitivity kit.
Quantification is not completely reliable due to adapter dimers and unextended primers.

Libraries are quantified by QPCR analysis:

- Dilutions of a library with known sequencing efficiency are used to build a standard curve.
- Quantify only amplifiable molecules.
- High correlation with cluster number.

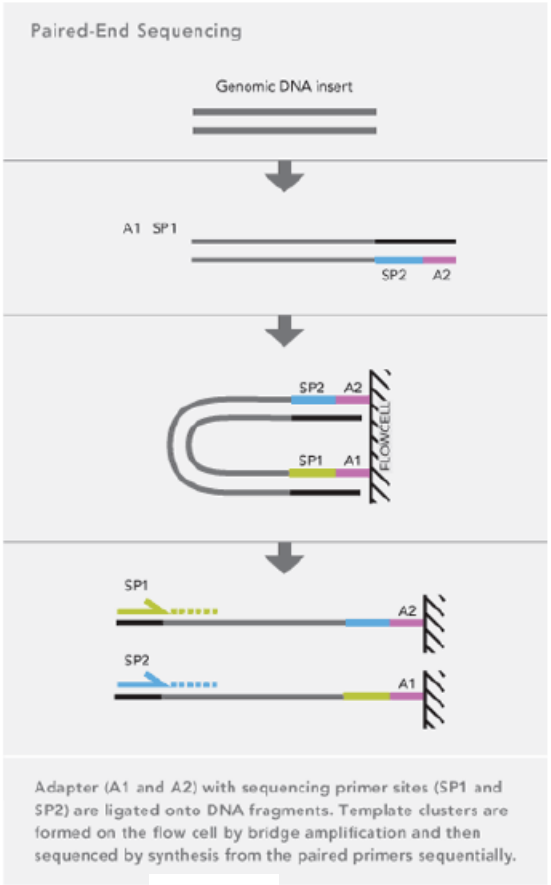
Meyer *et Al. NAR.* 2007
Quail *et Al. Nat meth.* 2008

GENOME

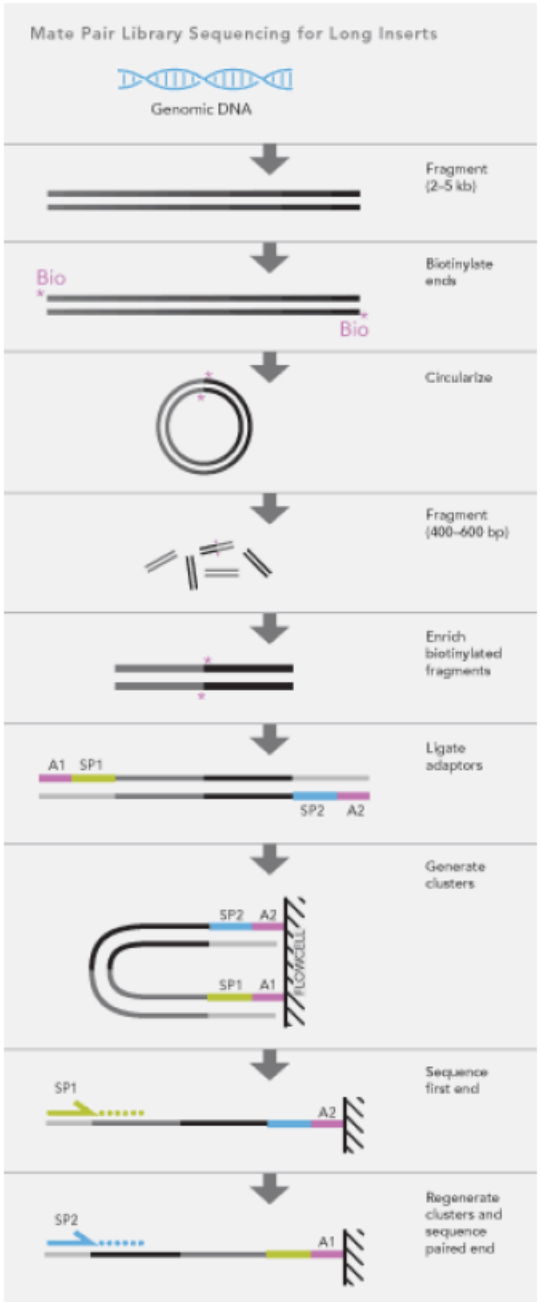
Library types

Single read sequencing

Paired-end sequencing (up to 500 bp insert)

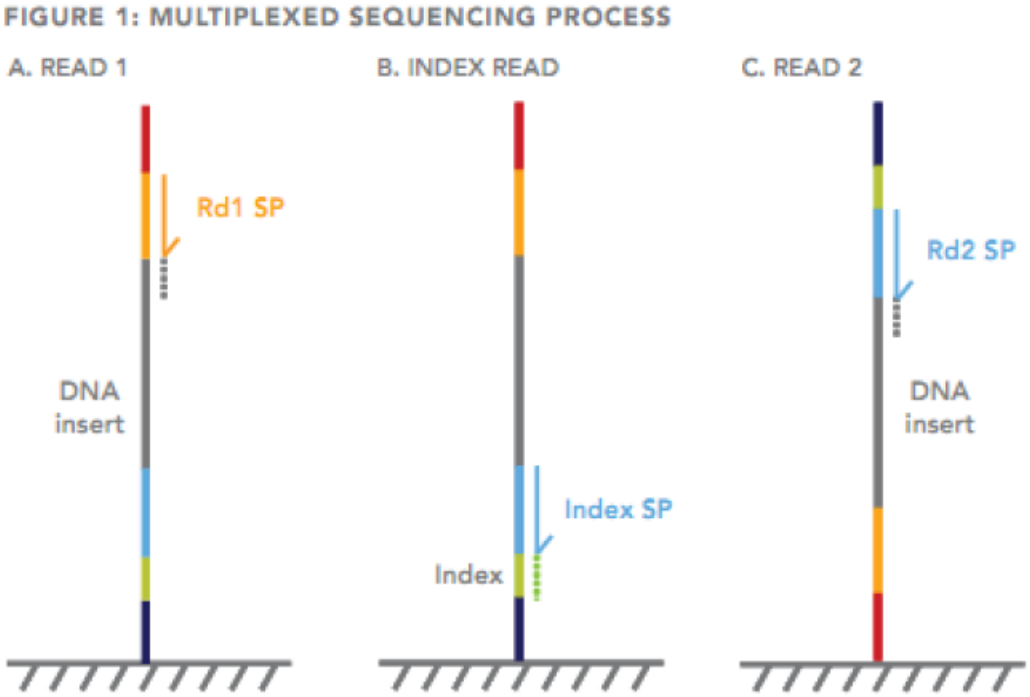


Mate Pair library sequencing (2-5 Kb insert)



GENOME

Multiplexed sequencing

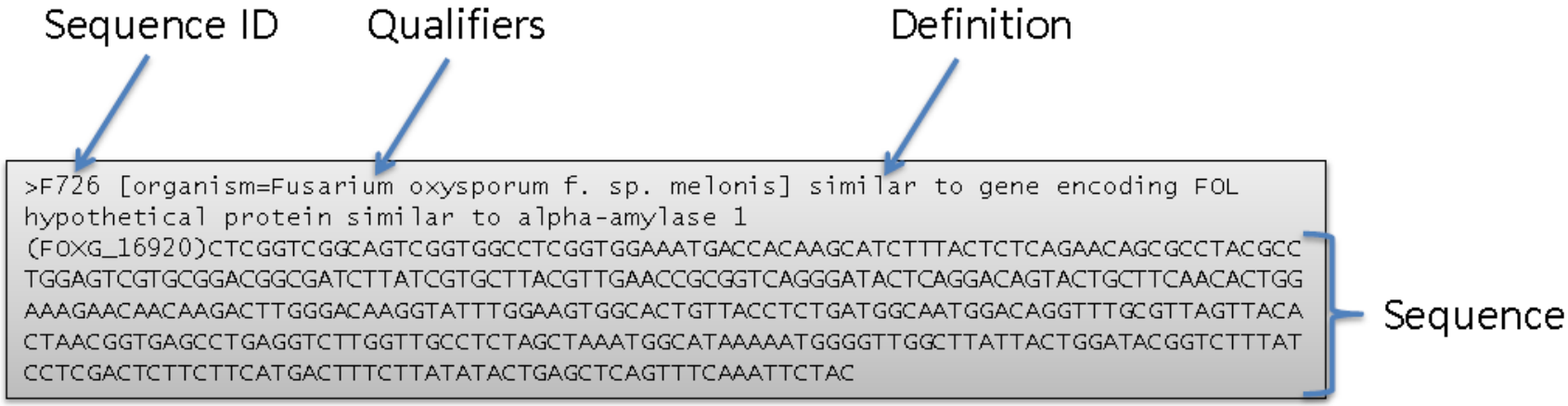


Up to 12 samples can be mixed and sequenced in each lane.

Sample multiplexing involves a total of three sequencing reads, including a separate index read.

GENOME

FASTA format



This is a general sequence format. Only sequence, no sequencing quality information

GENOME

Phred quality score

- Based on the analysis of peaks around each called base.
- Ranges from 4 to 60 (higher values express higher quality).
- Quality scores are logaritmically linked to error probabilities:

$$Q = -10 \log_{10} p$$

Phred quality score	Probability that the base is called wrong	Accuracy of the base call
10	1 in 10	90%
20	1 in 100	99%
30	1 in 1,000	99.9%
40	1 in 10,000	99.99%
50	1 in 100,000	99.999%

http://www.phrap.com/phred/

GENOME

Illumina qseq format

HWI-EAS221	2	3	56	0	1200	0	1	.ACCCGGGGAAGGGTGGGGGC.GCCCTTGGTGTCGC	DGRIIDEGJJDDDDJCGEDDDDDDIDIJKDEMDPDDD	0
HWI-EAS221	2	3	56	0	1211	0	1	.ATTTTTTCATACACATTAAC.ATTTTTTTTCTTTC	DMDPVUQHHDDDKNIKDOHDDDHDIKHDMNLEHKLI	0
HWI-EAS221	2	3	56	0	1219	0	1	..CCCGGATGGCCCAGTC.CT.CAAACCGCTCTGTG	DEKDGDDDDDDGMDDKDJGDJDDDIIDKKKDMDNDID	0
HWI-EAS221	2	3	56	0	1325	0	1	.TTTTGGGGGGTTGTCTTTCT.GTTTTTGCTCTGTG	DLUPSRKHDKOFMHNDWOWLHDMRZWWTFKODNDND	1
HWI-EAS221	2	3	56	0	1374	0	1	.AATTTTTTTTACACTTTTTT.TTTTTTCCCCCCCC	DORFR[[>SKDDDHIDHKKMIIDMDHDHHDDLHKHNK	0

Machine Run Lane Tile X.coord. Y.coord Index Pair Sequence QualityScore Filtering

Illumina preliminary format: includes all called sequences and a quality filtering flag.
Is the file format processed from the illumina
Sequences with flag=0 will be then removed in the final fastq-like format.

GENOME

Read Segment Quality Control Indicator

Lowest quality which is found in a Illumina file is in fact B (Q=2) and it has a special meaning:

- *At the ends of some reads, quality scores are unreliable. Illumina has an algorithm for identifying these unreliable runs of quality scores, and we use a special indicator to flag these portions of reads.*
- *A quality score of 2, encoded as a "B", is used as a special indicator. A quality score of 2 does not imply a specific error rate, but rather implies that the marked region of the read should not be used for downstream analysis.*
- *Some reads will end with a run of B (or Q2) basecalls, but there will never be an isolated Q2 basecall. (Tobias Mann of* Illumina*)*

```
@ILLUMINA-C3C24B:2:1:1:30#0/1
CAAACTCCCCAGCCACGACCACGTCGGAGGTCCTGA
+ILLUMINA-C3C24B:2:1:1:30#0/1
a\O]a`^^_WZW[\XULVW]WBBBBBBBBBBBBBBB
```

Read Segment Quality Control Indicator

GENOME

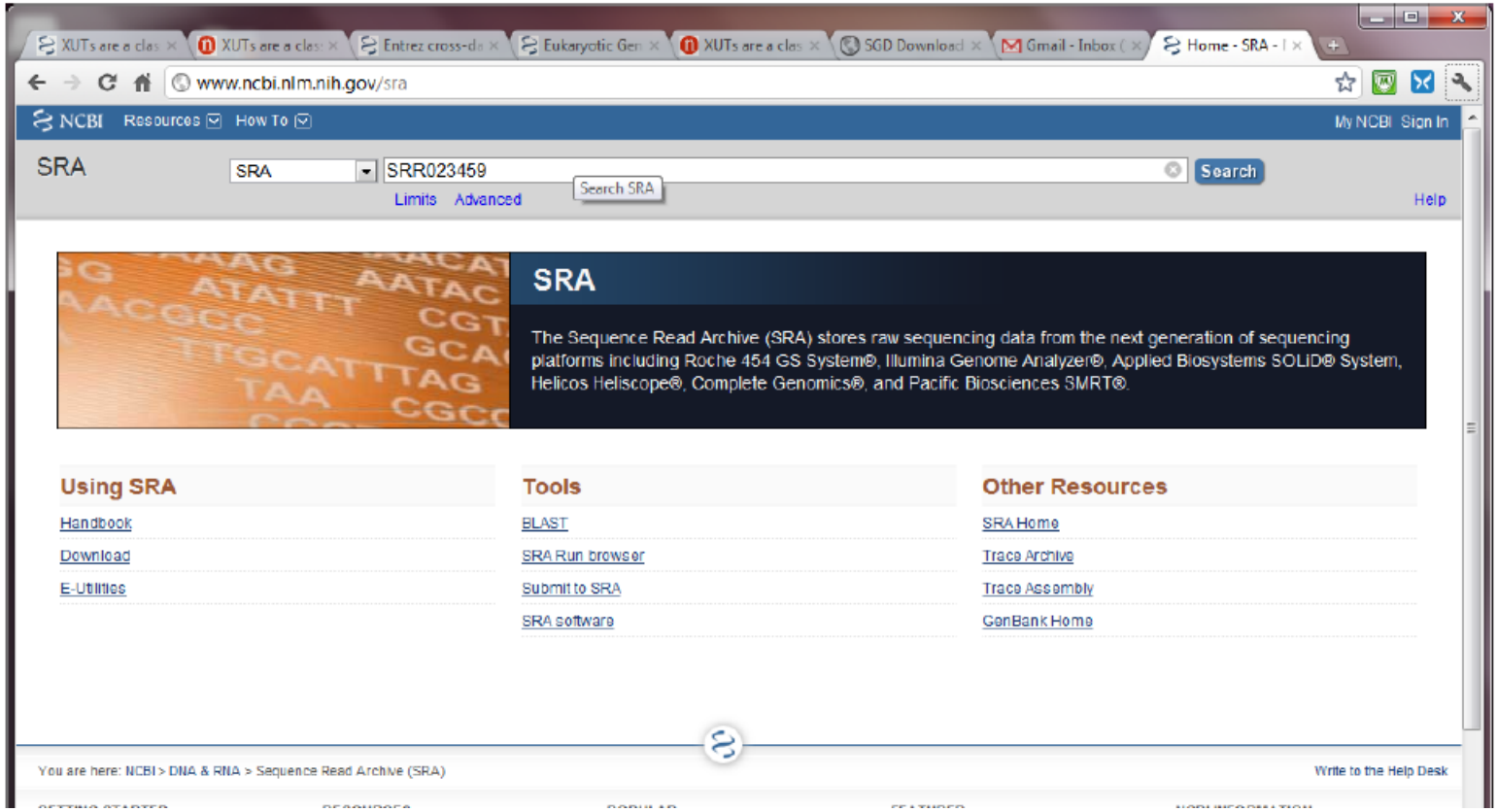
How to get data from public repositories

OBTAINING PUBLIC DATA

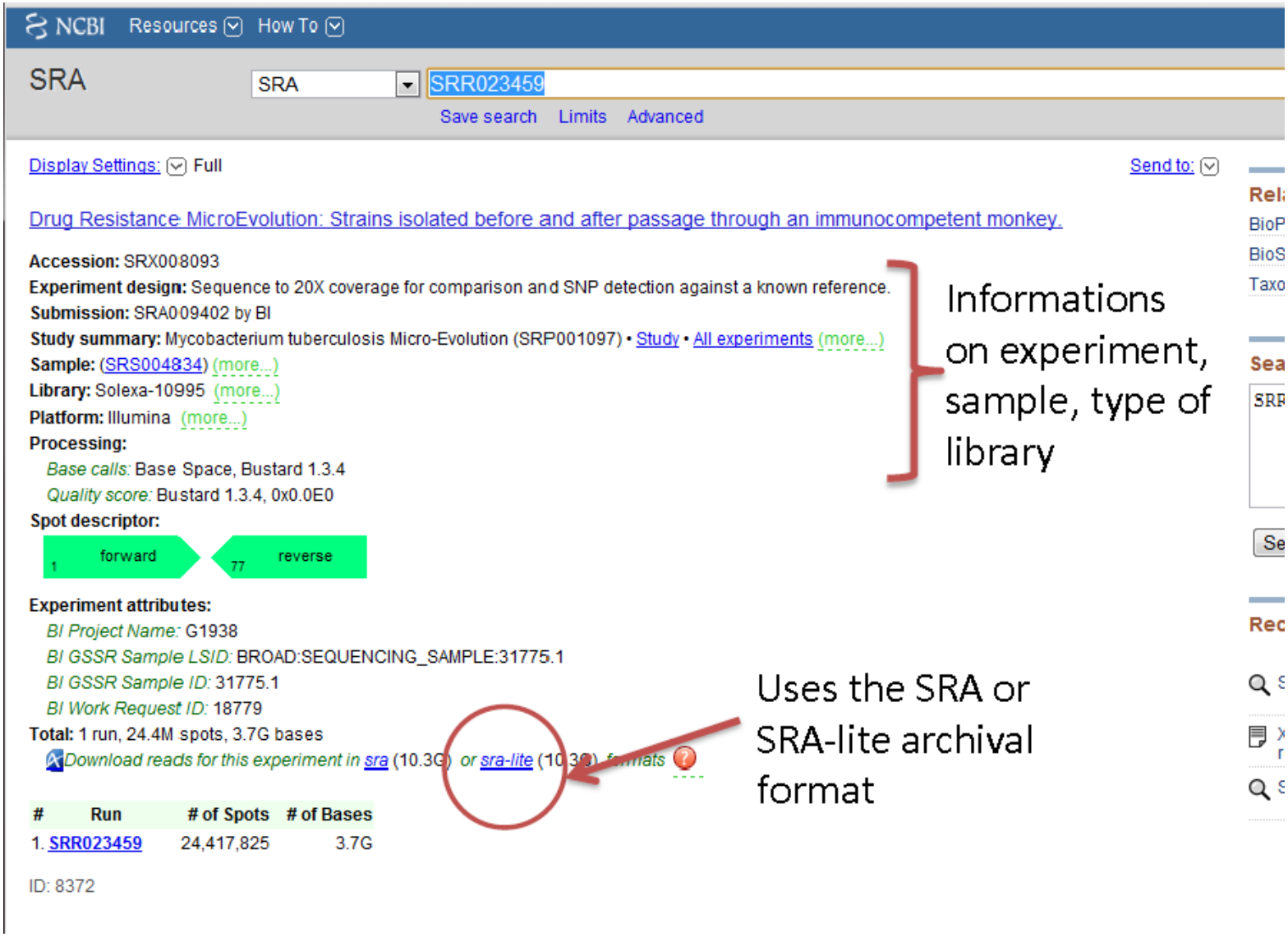
GENOME

Short Reads Archive

www.ncbi.nlm.nih.gov/sra



GENOME


NCBI Resources How To
SRA
SRA
SRR023459
Save search Limits Advanced
Display Settings: Full
Send to:
Drug Resistance MicroEvolution: Strains isolated before and after passage through an immunocompetent monkey.
Accession: SRX008093
Experiment design: Sequence to 20X coverage for comparison and SNP detection against a known reference.
Submission: SRA009402 by BI
Study summary: Mycobacterium tuberculosis Micro-Evolution (SRP001097) • Study • All experiments (more...)
Sample: (SRS004834) (more...)
Library: Solexa-10995 (more...)
Platform: Illumina (more...)
Processing:
Base calls: Base Space, Bustard 1.3.4
Quality score: Bustard 1.3.4, 0x0.0E0
Spot descriptor:
1 forward 77 reverse
Experiment attributes:
BI Project Name: G1938
BI GSSR Sample LSID: BROAD:SEQUENCING_SAMPLE:31775.1
BI GSSR Sample ID: 31775.1
BI Work Request ID: 18779
Total: 1 run, 24.4M spots, 3.7G bases
Download reads for this experiment in sra (10.3G) or sra-lite (10.3G) formats
Run # of Spots # of Bases
1. SRR023459 24,417,825 3.7G
ID: 8372
Informations on experiment, sample, type of library
Uses the SRA or SRA-lite archival format

Genome bioinformatics 2

What bioinformaticians do today…

November 9, 2016

Genomics problems 1

(two extremes according to biological nature)

- Genome not known: assembly from reads (determine the genome)

- Genome well known: Assembly of reads to genome (look for mutations)

Genomics problems 2

(two extremes according read length)

- Long reads (Sanger sequencing – traditional) Laborious and accurate. Not used for whole genomes (needs subcloning). Useful for final verification of mutants.

- Short reads (NGS, next generation sequencing): Current standard for almost everything. Very high coverage is needed for difficult problems

Genomics problems 3

Algorithmic issues, data representations

- Reads are sequences. Traditional alignment is too slow.
- Network of reads = graphs. Nodes = sequences, edges = “similarity”. If “Similarity” = overlap sequence, we have de Bruijn graphs.
- Hamiltonian paths, circles: each node visited once. NP complete
- Eulerian paths, circles: each edge visited once. Complexity $O(E)$ - $O(E^2)$
- Running times are best if we use a) de Bruijn graphs with a subword representation of overlaps and/or b) Clever representations for the reads (Burroughs Wheeler transform) and c) Clever indexing allowing fast search (Ferragina).

ASSEMBLING STRATEGIES 2

Assembling short (next generation sequencing) reads

Sequencing (assembling) strategies

- Determine subsequences of the genome
- Find overlaps and assemble overlapping reads
- Traditional (hierarchical) way: long and accurate subsequences from subclones, assembly by exhaustive sequencing (Smith Waterman)
- Next generation sequencing: Shotgun sequencing of the whole genome, assembly by graph-based methods.

G1

Changing technology

Human genome (2001):

~30 million reads

Up to 800bp long

→ 7.5X coverage

Panda genome (2009):

~2.7 billion reads

Average of 50bp

→ 56X coverage

Only short read methods used today….

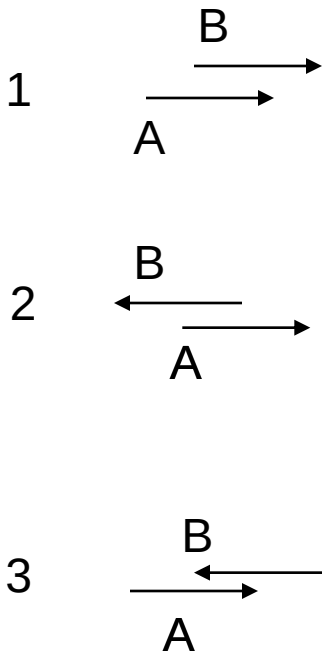
GENOME

De Bruijn Graphs: an introduction

- A graph whose nodes are sequences of symbols from some alphabet and whose edges indicate the sequences which might overlap. (Wolfram MathWorld)

- Given a set of sequences, a de Bruijn graph might be built looking for all the possible connections between the sequences.

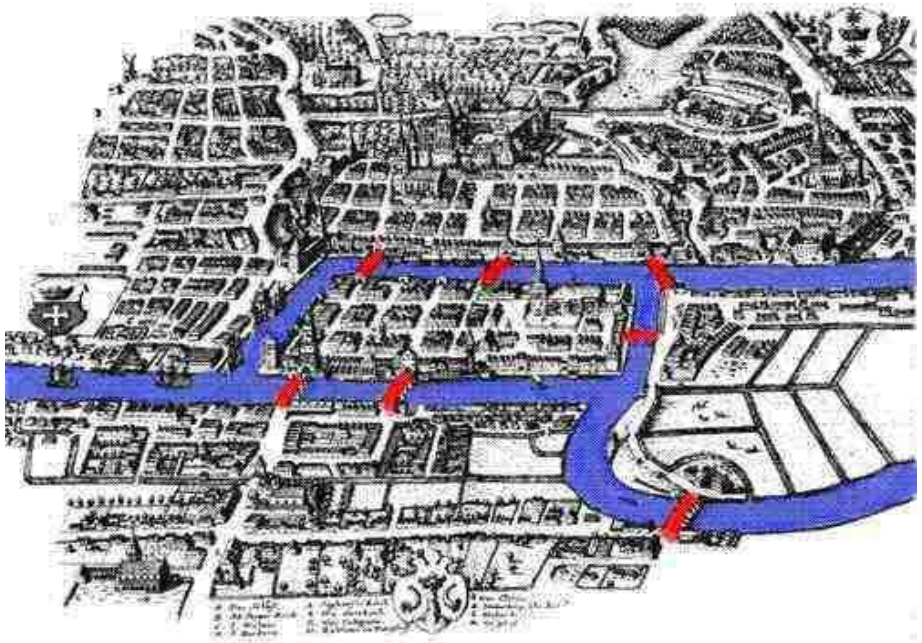
- Usually, the graph is not built looking for overlaps of whole sequences, but rather from their subsets. **(words)**



Overlaps between sequences can be of a very few types: unidirectional (1), divergent (2) and convergent (3). Containment (e.g. A in B) or identity (A=B) can be omitted without loss of info

G1

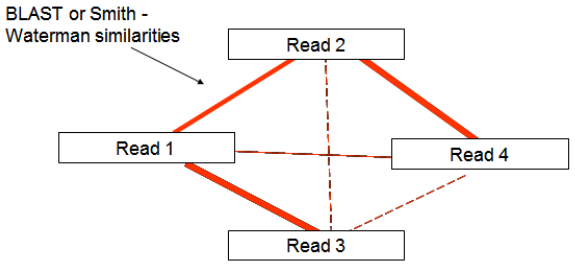
- In graph theory, an Eulerian trail (or path) is a walk in a graph which visits every edge exactly once. Similarly, an Eulerian circuit or Eulerian cycle is an Eulerian trail which starts and ends on the same vertex.



Graphs (networks) in sequencing

- We now link the idea of graph tools in sequencing to other network ideas (like the protein universe in previous lectures)

Practical solutions sequence read graphs



- Originally, the edges were determined by Smith-Waterman.
- The overlap sequences were used to reconstruct the sequence as a jigsaw puzzle.
- This is feasible with long and accurate reads (Sanger sequencing of subclones), and was done for the human genome USING very large human and computational costS.
- The same can not be done for short reads that contain many errors (next generation sequencing)

The idea of overlap graphs was abandoned IN THE PAST because SW is expensive and it is very difficult to build a network. Now we use it again…

The Scream

Problems:
Alas, the problem is NP-hard!

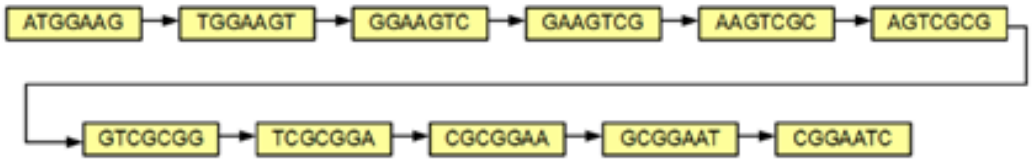
- The genome (from which the reads come) is a Hamiltonian path in the graph.
- Finding a Hamiltonian path is an NP-hard problem.
- But, we can find an alternative representation of the graph where we will look for Euler paths, which are not NP hard but $O(E)$ - $O(E^2)$.

Pevzner et al.

The way out 1: De Bruijn graphs

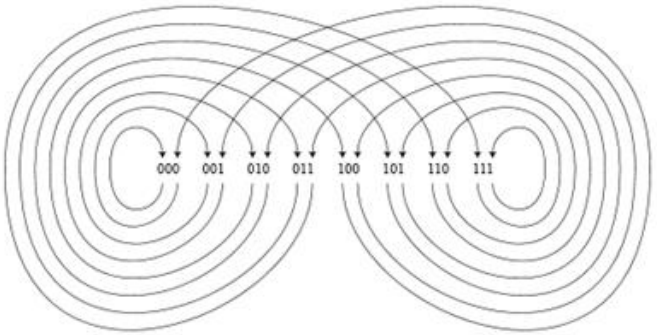
"k-mer network"

- De Bruijn graphs in mathematics are built from sequences of the same length ("k-mers") from a long text (In bioinformatics: "sliding window").
- Each k-mer is connected to the next window. This gives a directed graph.
- The graph has one unique long path: the text itself, i.e. the genome..
- So, we can use relatively inexpensive approximations to finding the genome string (Euler walk finding)
- Finding an Euler walk is not NP hard, complexity is proportional with the sum of the numbers of edges and nodes. (The equivalent Hamiltonian problem would be NP hard)



A more recent idea
that worked

De Bruijn sequence



An Eulerian path in a de Bruijn graph yields a de Bruijn sequence

A sequence in which every k-mer occurs exactly once

B(2, 3) = 00010111

K-mer

- In practice, a sequence is broken up into different substrings of varying length, called **k-mers**.

- The graph is then built from these k-mers, trying to connect each original string to another.

- As it would be unfeasible to look for all the connection among all possible **k-mers** of our dataset, a **minimum size** is specified. This parameter is often indicated by the letter **k.**

- Once all possible nodes have been determined, we “walk” the graph going from one sequence to another through the various nodes. When we reach a sequence with no outgoing nodes, we have determined our “superstring”, or **contig**.

K-mer and Scaffolding

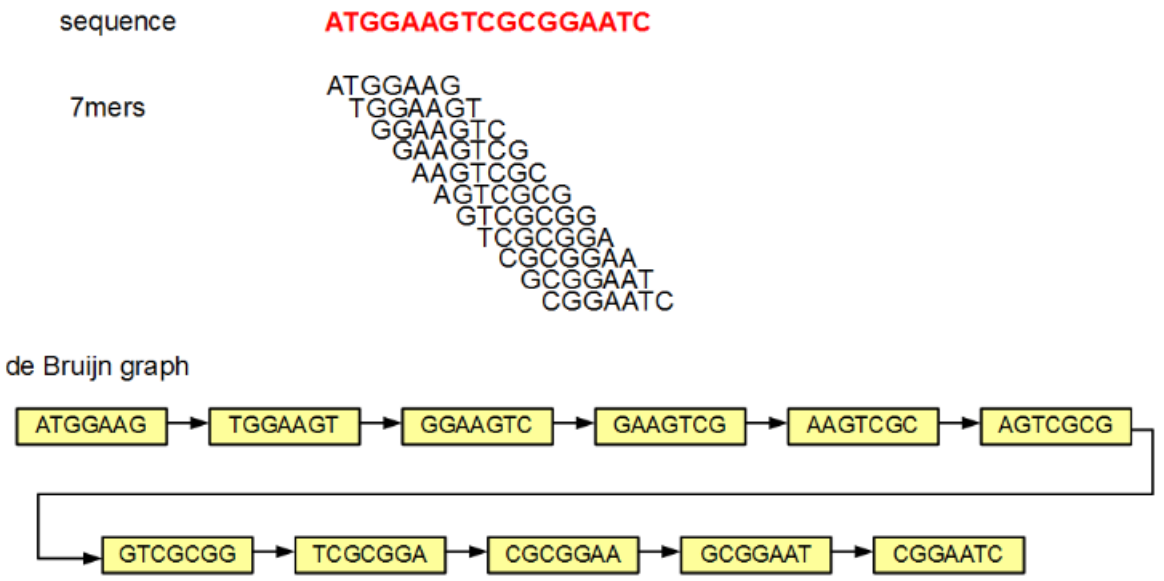
- Different choices of the K value return different assemblies. It is not possible to know in advance which K value will give the best result; it must be determined empirically by performing different assembly experiments.

- Lower Ks increase sensibility but decrease specificity, and require greater memory. They require to consider only mismatches when comparing substrings.

- Higher Ks produce smaller, more specific graphs. However, they are less sensible, and might require to consider also possible gaps.

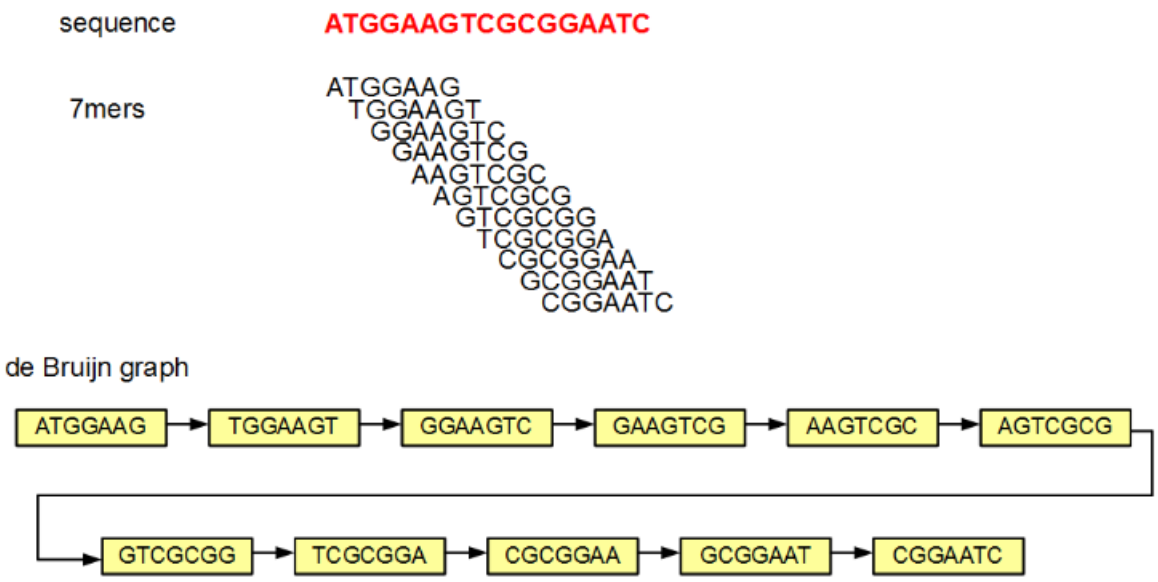
- Using the information provided by different libraries (e.g. a different dataset of longer reads, or the paired-end information) it is possible to **scaffold** our contigs into longer sequences.

De Bruijn graph assembly



A sequence or sequences supports a subgraph of the k-order de Bruijn graph

De Bruijn graph assembly



Finding a Eulerian path reconstructs the original sequence

De Bruijn graph assembly

By enumerating every k-mer in every read, we can simulate sequencing by Hybridization

Sequencing by hibridization was an old suggestion that could not be built into a working sequencing system….

The Scream

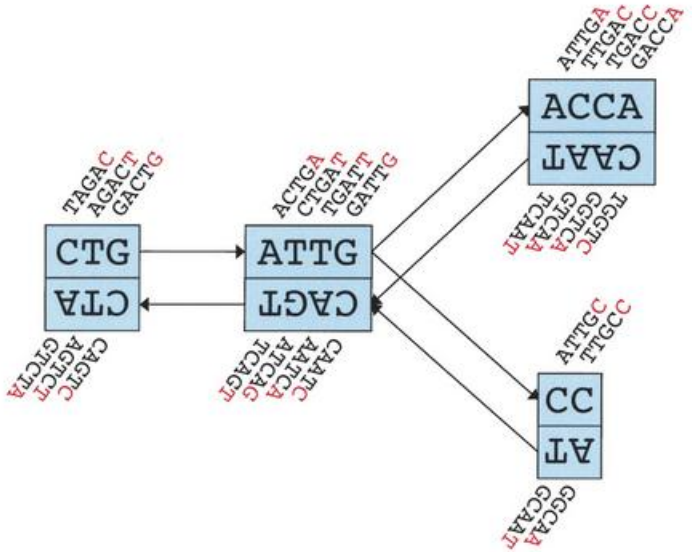
De Bruijn graphs: Alas, the way out is almost lost!

- Reads contain errors
- Overlaps can be very short, at times even missing..
- Reads from different strands
- Repeats in genomes (eukaryotes)
- Missing sequences (that result in scaffolds)
- Huuuuuge numbers of reads

All this makes the use of graph algorithms difficult…

Strand ambiguity

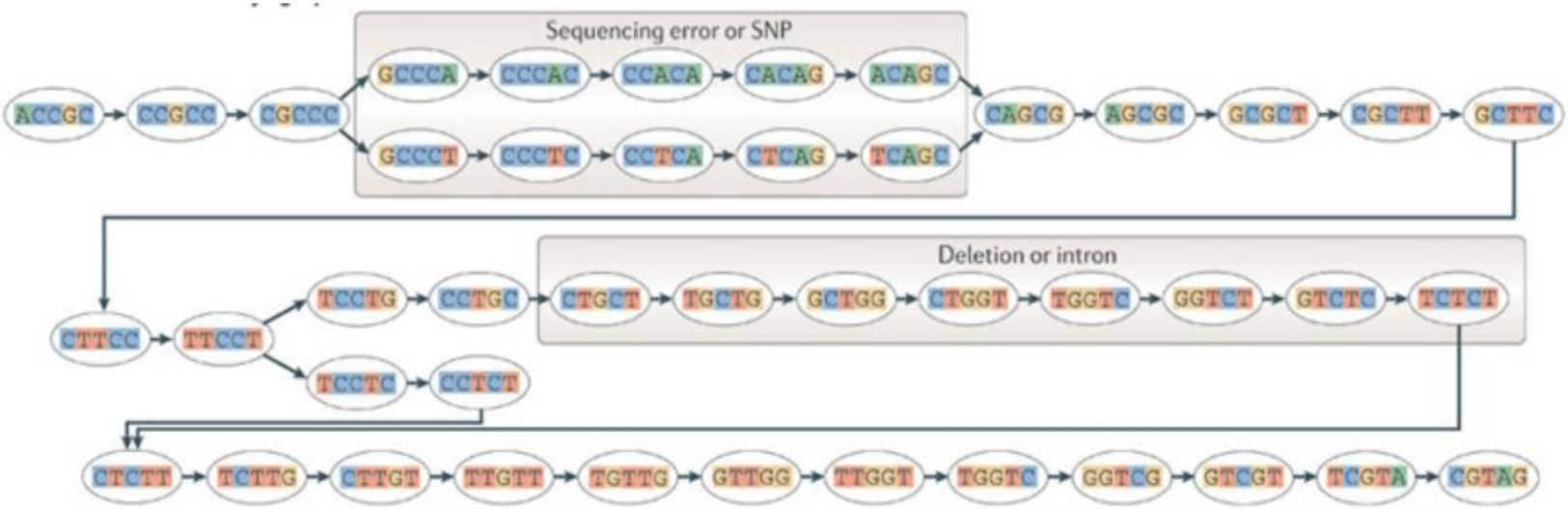
Typically sequencing is not strand-specific



Solution: treat a k-mer as equivalent to its reverse complement

Read errors

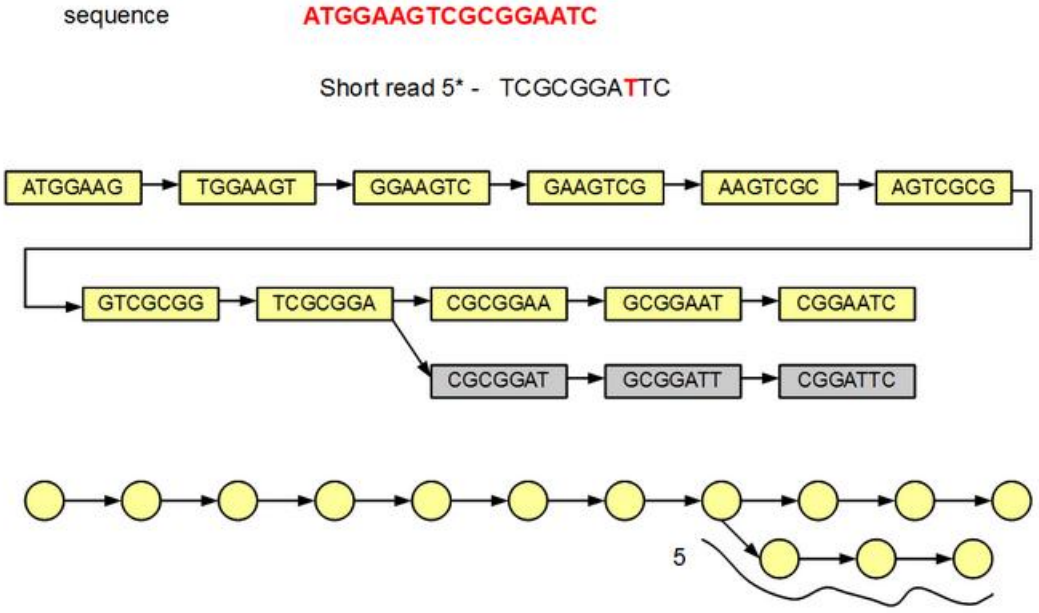
Reads are not error free.



Complexity of the graph can explode.

Filtering read errors

In practice k-mers are **counted**

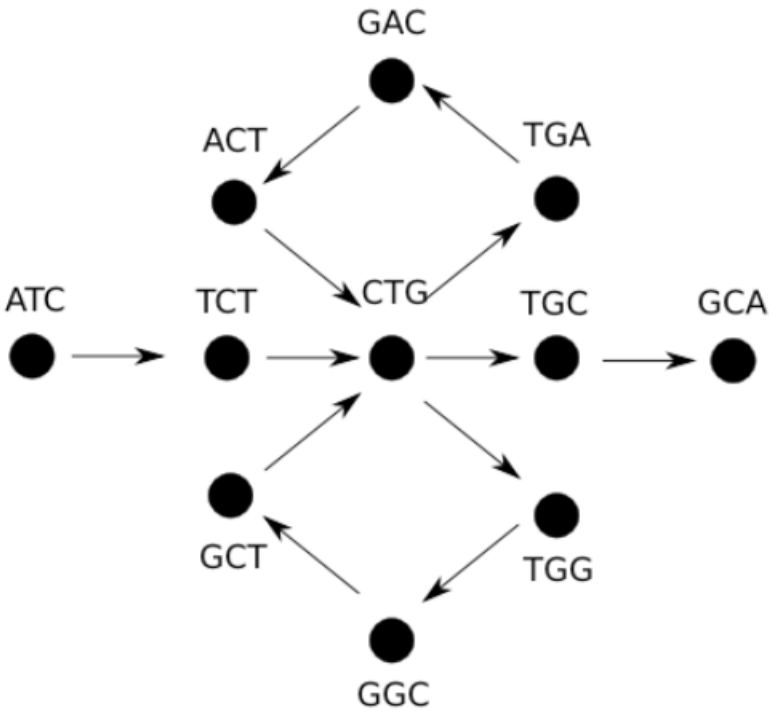


Low weight paths can be pruned.

Eulerian path ambiguity

Reads

ACTG
ATCT
CTGA
CTGG
CTGC
GACT
GCTG
GGCT
TCTG
TGAC
TGCA
TGGC



Genome Reconstructions

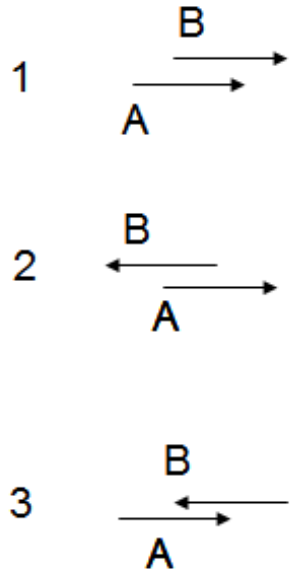
ATCTGACTGGCTGCA
ATCTGGCTGACTGCA

Eulerian paths are not always unique!

De Bruijn graphs: There is still a way out!

- In bioinformatics, we know the kind of overlaps and the kinds of errors.
- All these can be expressed in graph theory terms, e.g. weighted graphs, bidirectional graphs, etc.
- So we can extend the concept of de Bruin graphs to networks in which path-finding is still feasible (time, memory)
- It works, mostly for bacterial genomes (haha)

"k-mer network"



Overlap types

K-mer size tradeoff

Large k: less ambiguity, worse coverage.

Small k: more ambiguity, better coverage.

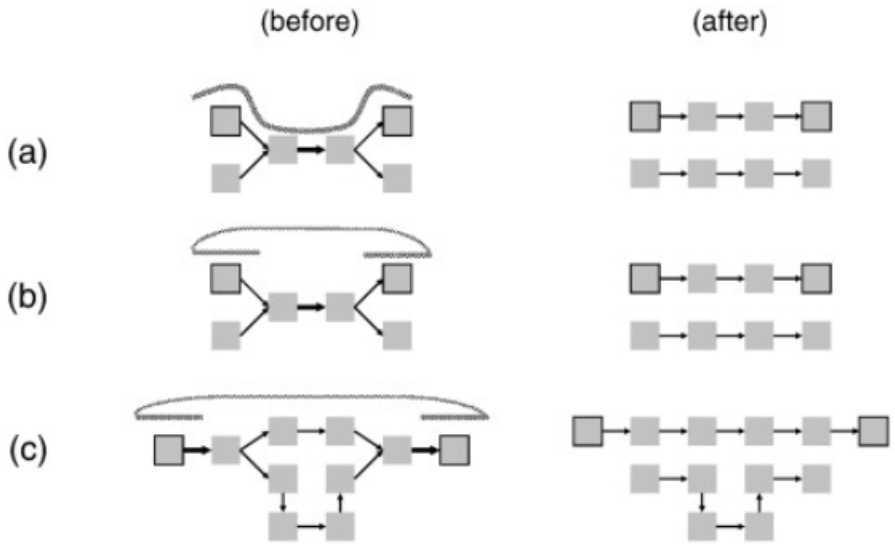
In practice: k = 20-50 is used.

One possible strategy: use multiple k-mer sizes

Pevzner: SPADE program

Scaffolding

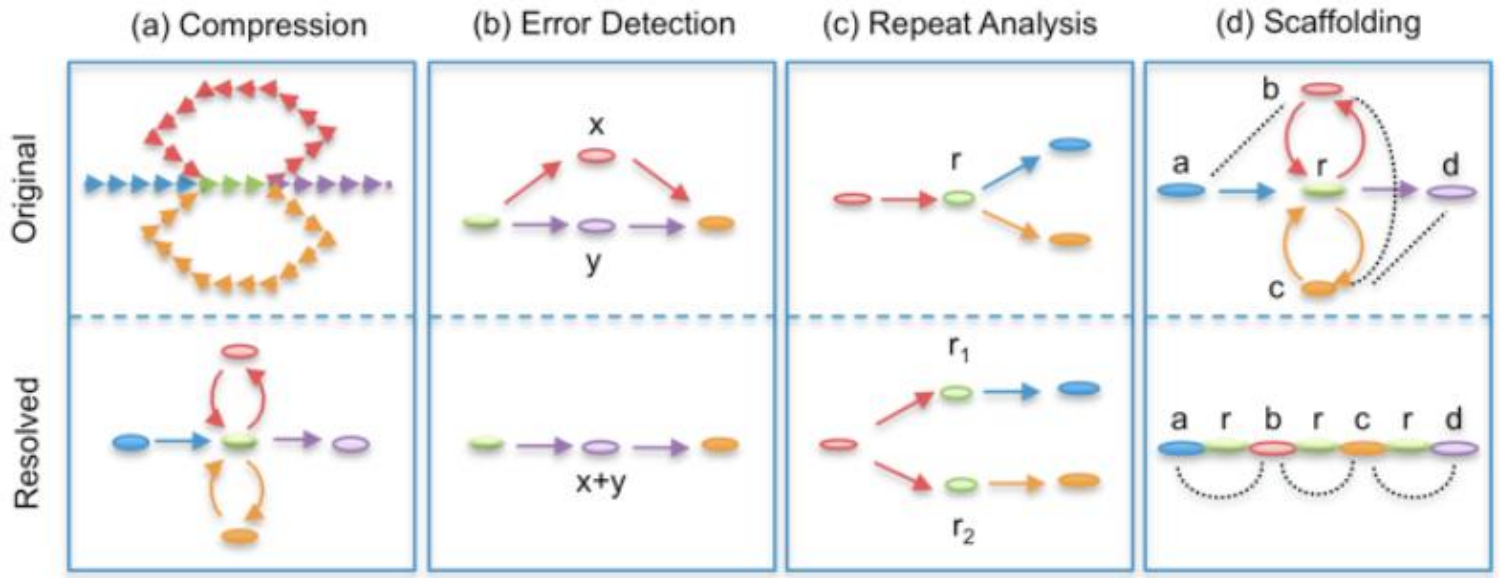
Common strategies involve using full-length reads to resolve ambiguity.



Either threading a read through the graph (a), or using mate-pair distance information (b, c)

De Bruijn graph heuristics

In Summary



De Brujn Assemblers - an overview

	VELVET	ABySS	SOAPdenovo	CLC
Read filtering	NO	NO	YES	YES
Scaffolding	YES	YES	YES	YES
Long Reads	YES	NO	NO	YES
Memory		Distributed	Parallel	Parallel
Source code	**GNU**	**GNU**	**Executable**	Commercial

This was last year. Now we use SPADE!

DeBruijn assemblers

- SPADE (practicals)

New trick: back to overlap graphs

- There are new, compressed data structures [1] that allow the building of overlap graphs in practice
- SW is not necessary, graph is built directly from the index.
- No need for k-mer (subword) decomposition (like with a de Bruijn graph)
- Path building is feasible (using the same tricks as before: collapsing reads, strand ambiguity, repeats, scaffolding etc…)
- String Graph Assembler of Simpson and Durbin.

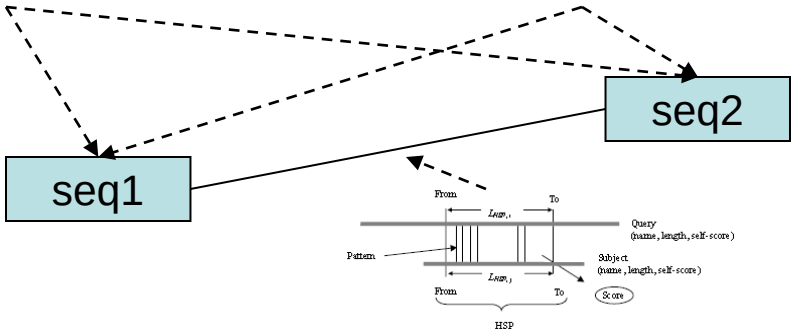
[1] Burrows Wheeler transform and Ferragina-Manzini index (FM), more efficient than hash tables… see later lectures

Why are there so many different network representations?

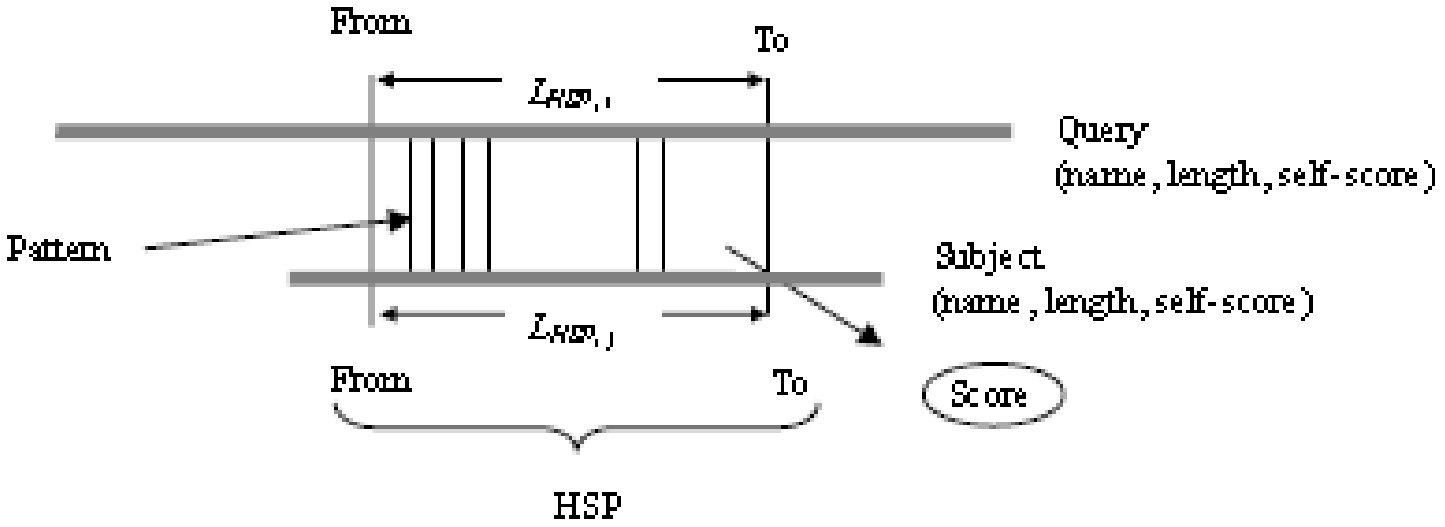
- Networks are Entity-Relationship models (the most complex models of knowledge representation)
- Nodes (sequences) can be mathematically represented as character strings, series of substrings, compositions, etc., etc..
- Nodes (sequences) can come from various experimental situations (genome sequencing, expression studies, etc, etc.)
- Edges are alignments that have a variety parameters (orientations, overlaps, scores, % identities, lengths, etc.etc)
- Some problems become tractable with certain representations only..

Various mathematical descriptions

Various experimental situations



Various alignment parameters


From
To
Query
(name, length, self-score)
Pattern
Subject
(name, length, self-score)
From
To
Score
HSP

Summary – what you should know

- Assembling genomes (or contigs) from reads is special problem composed of laboratory and computing tricks.
- Sequencing strategiew differ in the length and the accuracy of the reads.
- Early assembly solutions rely on accurate long reads (Sanger), exhaustive comparison (Smith Waterman or similar), and a jigsaw puzzle like assembly.
- Current solutions rely on large numbers of highly redundant and error-laden short reads (NGS)as well as network representations (De Bruijn graphs, overlap graphs) that avoid the need for direct comparisons such as SW.

An interesting De Bruijn example

The text (“genome”):

“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

Example courtesy of J. Leipzig and A. Clum

For the purposes of illustration, we can use human readable text to explore how assemblies work.
This is an example taken from Leipzig et al 2010.
In it he uses the opening paragraph from Dickens' "A tale of two cities".
It is an appropriate example because like genomes, it contains strings that are repeat over and over.

De Bruijn example

itwasthebestoftimesitwastheworstoftimesitwastheageofwisdomitwastheageoffoolishness…

Generate random ‘reads’ ⬇ ⬆ **How do we assemble?**

fincreduli geoffoolis ltwasthebe ltwasthebe 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 ltwasthebe 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

Traditional all-vs-all assemblers fail due to immense computational resources (scales with number of reads^2)

A million (10^6) reads requires a trillion (10^{12}) pairwise alignments

De Bruijn solution:

Represent the data as a graph (scales with genome size)

One concession in using human readable text as an example is that we can't keep the spaces. This string represents the genome we are going to sequencing and try to assemble. Sequencing is easy, we generate sub-strings at random from throughout the text. If this was Hi-Seq we'd have 10's of millions of these 'reads'. The difficult part is how we stitch them back together again in the right order.

An intuitive way to do this may be in all v all comparisons to search for overlaps. This is how traditional assemblers work. For datasets the size of hiseq this fails miserably. The solution offered by the De Bruijn approach is to represent the data as a graph.

De Bruijn 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

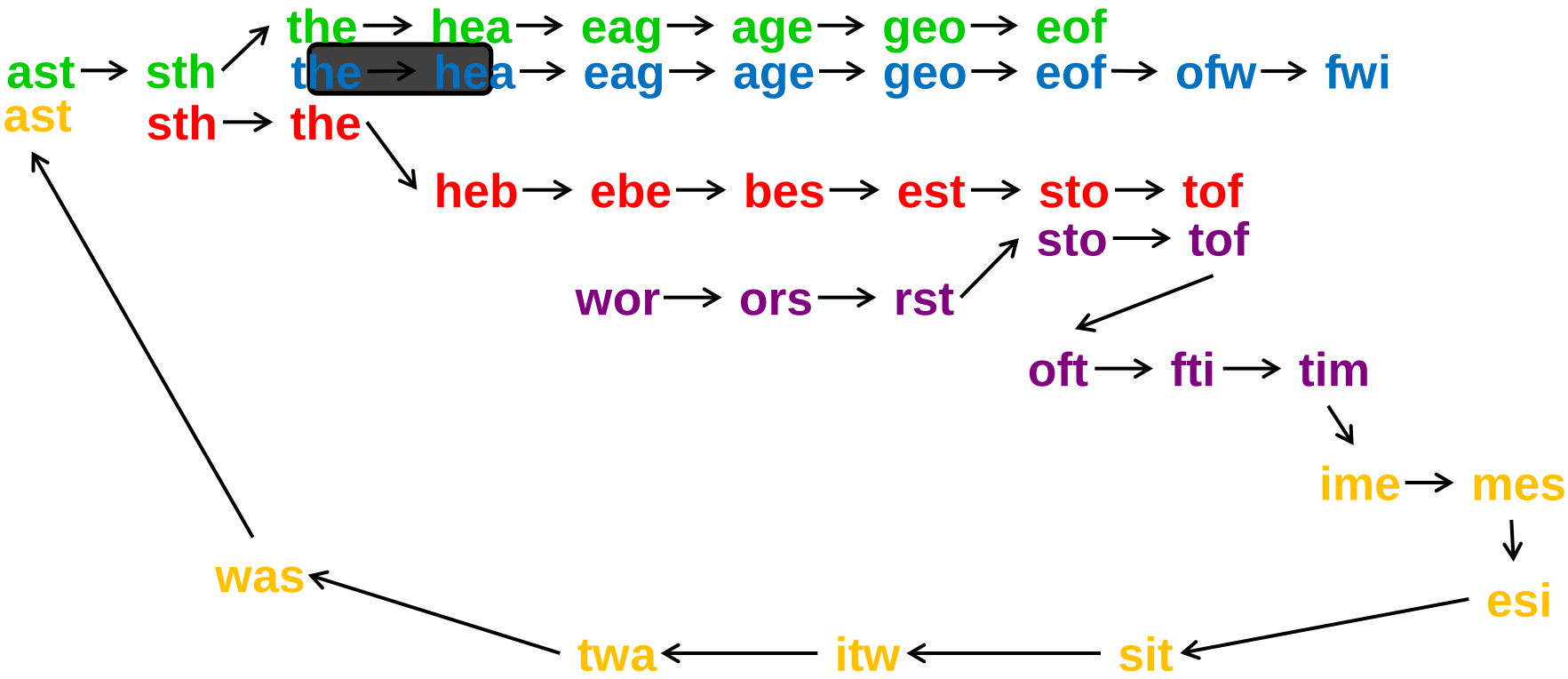
…..etc for all reads in the dataset

The first step of the De Bruijn assembler is to deconstruct the sequencing reads into its constitutive “kmers”.
A Kmer is a substring of defined length (in this case 3 letters).
To Kmerize the dataset, we move through our read in one letter increments from the beginning to the end untill we have recorded all possible 3 letter words.
We then do this for all reads in the dataset.
From this point on the algorithm operates on kmers rather than on the reads.

De Bruijn example

Step 2:

Build a De-Bruijn graph from the kmers

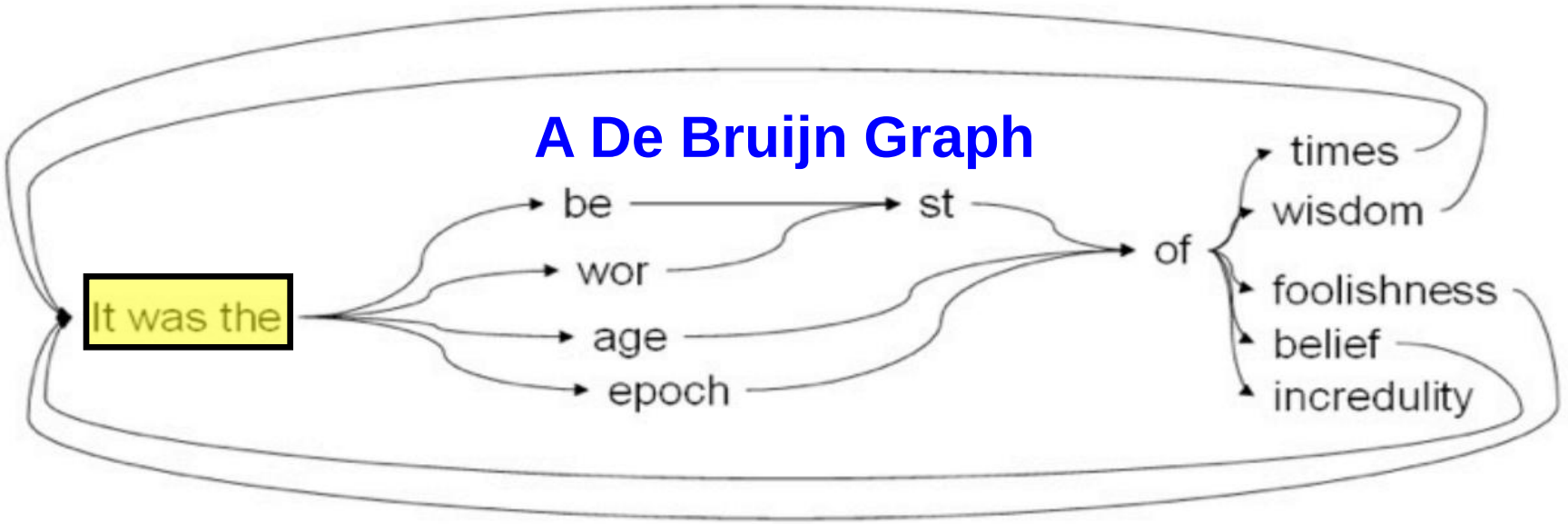


…..etc for all ‘kmers’ in the dataset

The next stage is to represent the stored "kmers" in the de druijn graph.
This is done by searching for overlaps of k-1 (in this case 2 letters).
-This is what the be druijn graph of a single read looks like. All consecutive kmers by k-1 bases. For Example the "he" from the first and second kmers.
-Adding kmers from a second read from an overlapping region of the genome shows how the graph can be extended. It also reveals redundancy in the data which need not be stored by the computer. This is how memory efficiency is achieved.
-Adding a kmers from a third 'read' that comes from a similar but non-overlapping part of the text illustrates the effect of repeats - We get a 'branch" in the graph.
-We this process for all kmers in the dataset.
-Long unbranched stretches represent unique sequence in the genome, branches and loops are the result of repeats.

De Bruijn example

Step 3:
Simplify the graph as much as possible:

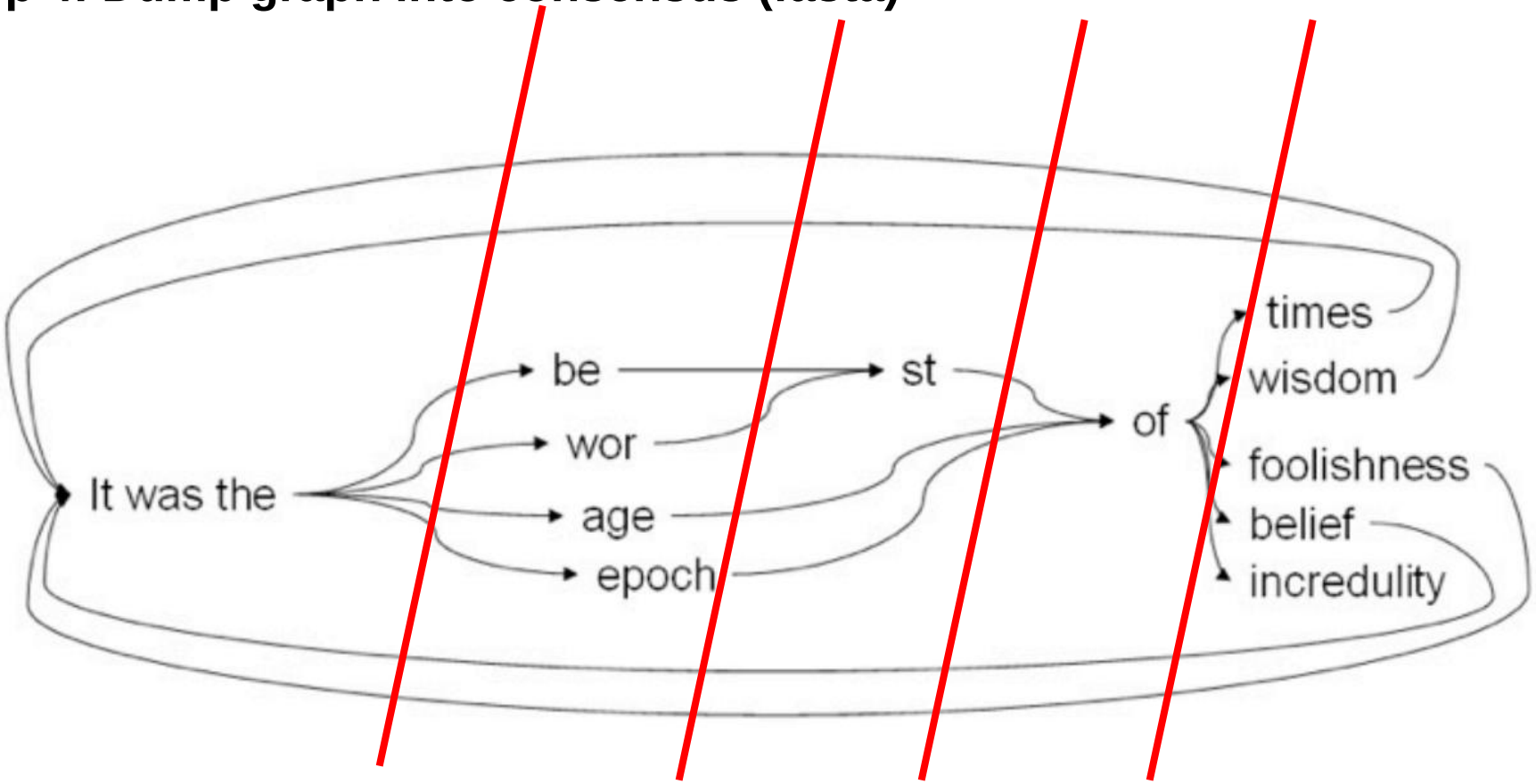


De Bruijn assemblies 'broken' by repeats longer than kmer

'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,.... "

Drawback of De Bruijn approach

Step 4: Dump graph into consensus (fasta)



No single solution!

Break graph to produce final assembly

The final step is to remove redundancy, result in the final De Bruijn Graph representation of our genome.

From this graph, we can see examples of both the strengths and weaknesses of this approach.

Strength is that the information from millions of reads is stored in computer memory in a graph that is proportional to the genome size.

Another strength is that the overlaps between reads are implicit in the graph, so all the millions v millions of comparisons are not required.

On the downside, information is lost as repetitive sequences are “collapsed” into a single representation.

Another problem is that while this may be a satisfying solution to a computational person, it is not practically useful to a biologist who wants to annotate genes etc.

Further information (the connectivity between contigs) is lost when we ‘chop’ the graph up into fasta sequences.