NGS DATA ANALYSIS

Dr Ben Dickins

NGS synonyms

Synonyms:

{second-generation, next-generation, ultra-high-throughput, massively parallel} sequencing.

Useful Reading

- Flicek, P., & Birney, E. (2009). Sense from sequence reads: methods for alignment and assembly. *Nature methods*. 6: S6-S12.
- Compeau PE, Pevzner PA, Tesler G. (2011) How to apply de Bruijn graphs to genome assembly. *Nature Biotechnology.* 29(11):987-91
- Narzisi G, Mishra B (2011) Comparing de novo genome assembly: the long and short of it. *PLoS One* 6(4): e19175
- Bradnam KR, et al. (2013) Assemblathon 2: evaluating de novo methods of genome assembly in three vertebrate species. *Gigascience*. 2(1):10
- The following fora are useful for seeking advice:
  - http://www.biostars.org (all bioinformatics)
  - http://seqanswers.com (NGS workflow)
  - http://stackoverflow.com (all computational)
  - http://wiki.galaxyproject.org/Learn (Galaxy)

Learning outcomes

- Grasp strengths and weaknesses of NGS platforms:
  - error rates and sources
- Appreciate future developments:
  - single-molecule sequencing
- Understand key computational concepts behind alignment and assembly:
  - Burrows-Wheeler Transform
  - de Bruijn graphs
  - contigs, scaffolds
  - N50, NG50
- Appreciate future developments:
  - Parallelisation/GPUs

Modes of analysis

application	what is sequenced	information sought	computation required
re-sequencing	one or a few individuals per barcode/lane	individual variants (sometimes in exome only)	alignment
re-sequencing	population per barcode/lane/flowcell	polymorphism	alignment
sequencing	one individual per lane/flow cell	genome (no reference)	assembly
metagenomics	population	new sequences (to be checked again many references)	assembly
RNA-Seq	individual per barcode/lane	transcriptome	assembly or alignment
ChIP-Seq	individual per barcode/lane	chromatin-bound sites	mapping/coverage

Many assemblers also offer reference-guided assembly

What you get back from most sequencers

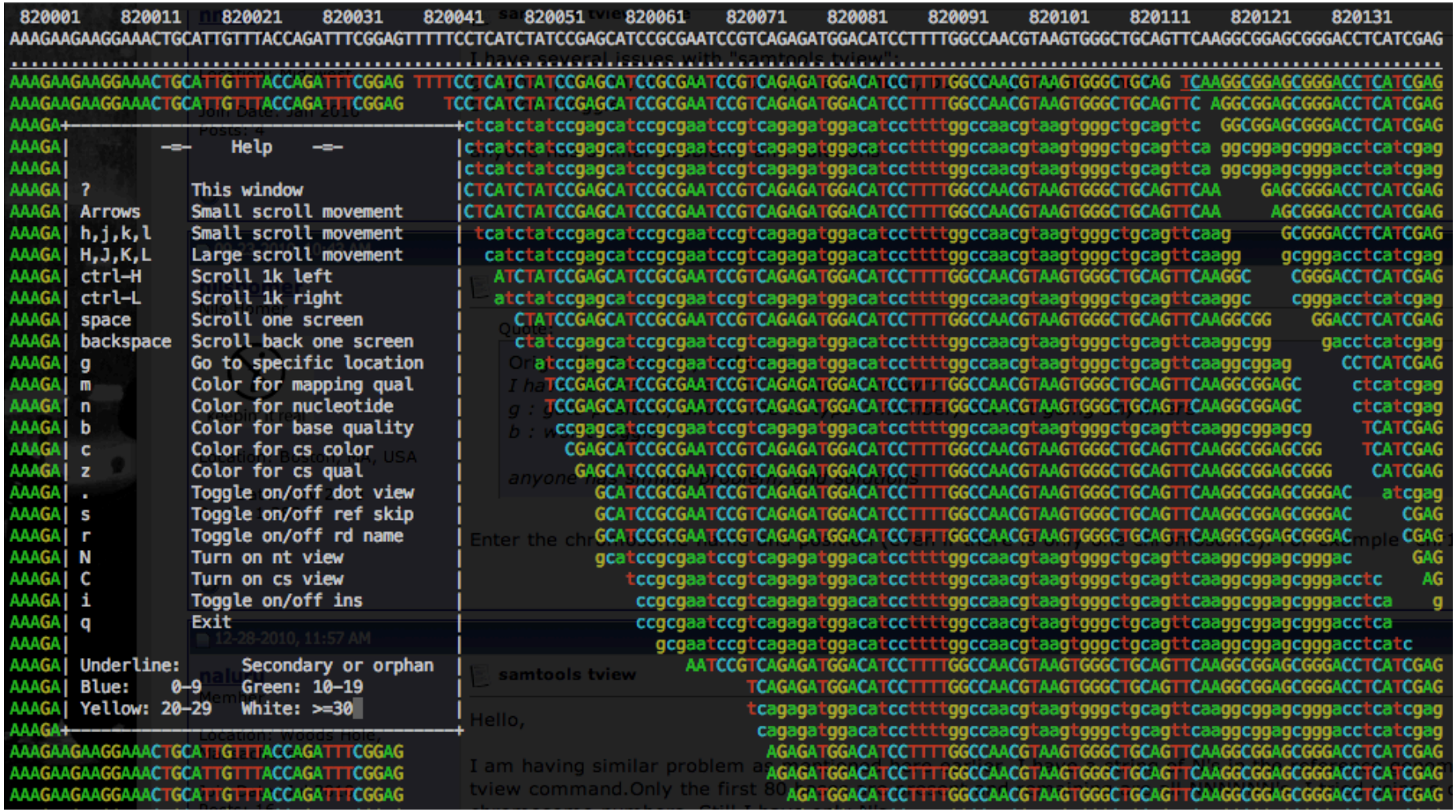
FASTQ = common to most NGS platforms:

```
@HWI-EAS209_0006_FC706VJ:5:58:5894:21141#ATCACG/1
TTAATTGGTAAATAAATCTCCTAATAGCTTAGATNTTACCTTNNNNNNNNNNTAGTTTCTTGAGATT
TGTTGGGGGAGACATTTTTGTGATTGCCTTGAT
+HWI-EAS209_0006_FC706VJ:5:58:5894:21141#ATCACG/1
efcfffffcfeefffcffffffddf`feed]`]_Ba_^__[YBBBBBBBBBBRTT\]][]dddd`dd
d^dddadd^BBBBBBBBBBBBBBBBBBBBBBB
```

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

The first line is the sequence (including ambiguous characters), the second line is the quality (Q, hex encoded). P = probability that a base is miscalled. In PE sequencing you get two files or an interspersed file (/2 after read ID). Check out Wikipedia for description: http://en.wikipedia.org/wiki/FASTQ_format (accessed 19/10/2014]

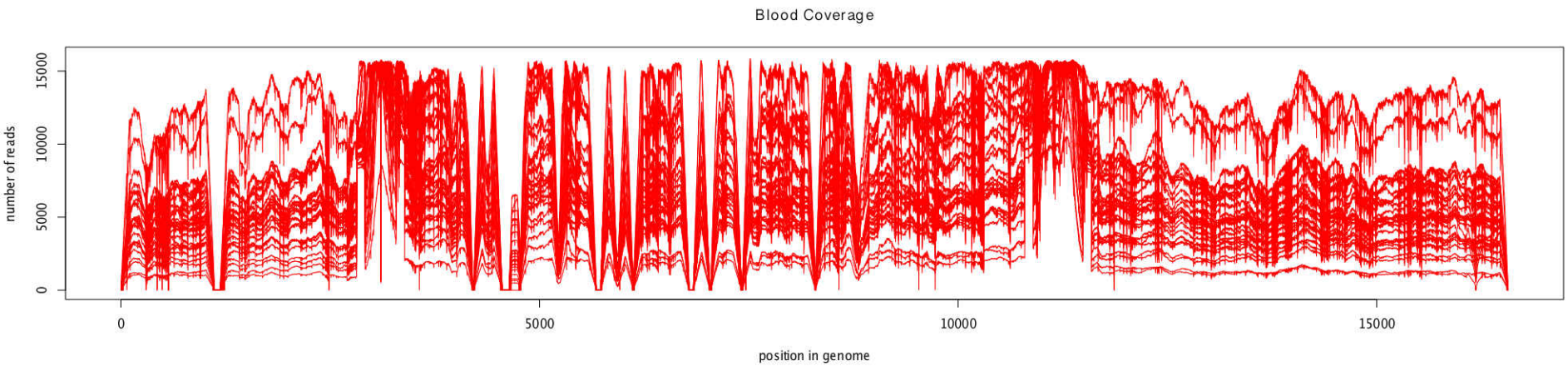
Alignment aka “mapping”

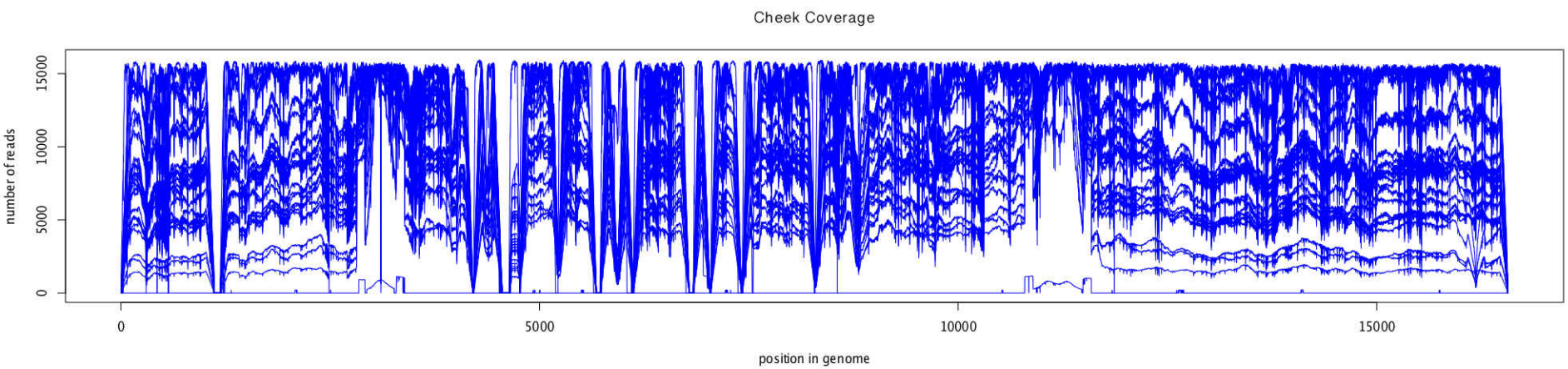


SAMtools tview

Key concept: coverage

How many reads cover each position in a genome?
Example from my research (multiple human mtDNAs):





Alignment methods

- Two alignment methods
- (i) Hash table–based implementations, in which the hash may be created using either the reference genome or the set of sequencing reads
- (ii) Burrows Wheeler transform (BWT)-based methods, which first create an efficient index of the reference genome assembly in a way that facilitates rapid searching in a low-memory footprint.

Alignment

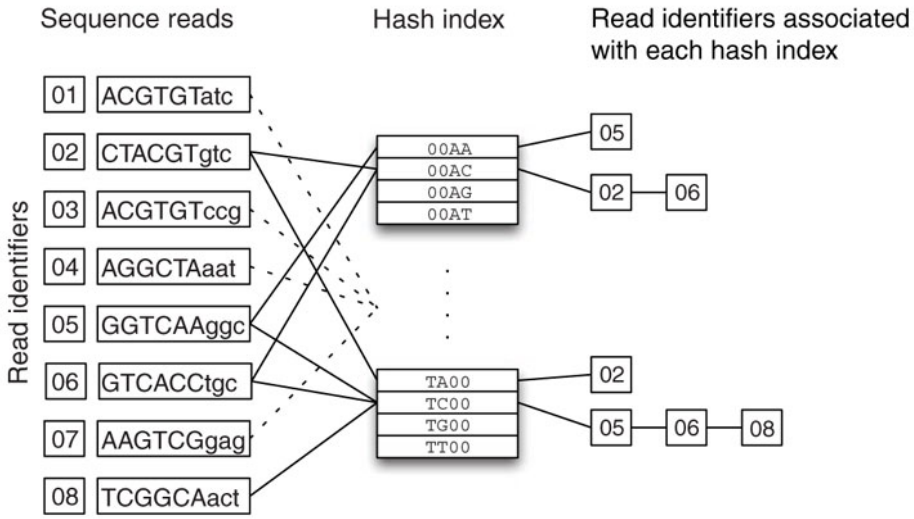
- **Multistep process**
- Rapid identification of a small set of places in the reference sequence where the location of the best mapping is most likely to be found, using heuristic techniques
- Slower/more accurate alignment algorithms (for example Smith-Waterman) are run on the limited subset

Hash Algorithms

- A hash table (aka dictionary) refers to a common data structure that indexes non-sequential data in a way that facilitates rapid searching.
- Hash algorithms build their hash table either on the set of input reads or on the reference genome.
- They then use the reference genome to scan the hash table of input reads (in the first case) or use the set of input reads to scan the hash table of the reference genome (in the second).

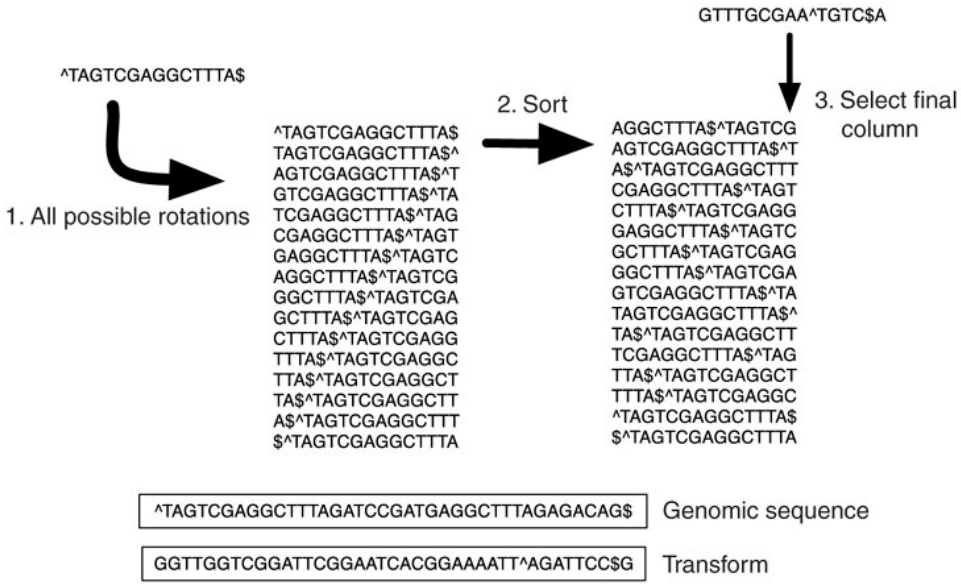
Bucket ← { key: value }

- Sequence reads with associated read identifiers with the regions that will be used for seed selection in capital letters and matched seeds of 0011 and 1100.
- Given read identifiers are associated with the seeds using a hash function (←; for example, a unique integer representation of each seed).
- Once such a hash table has been built for either the input read set or the reference genome, the corresponding data can be scanned with the same hash function, resulting in a much smaller subset of reads to more exactly align at each location in the genome.



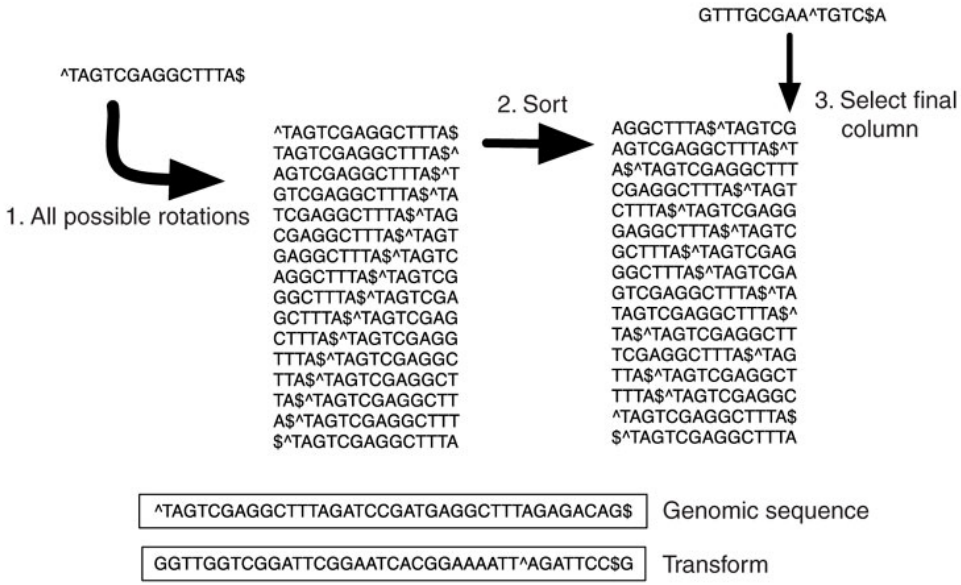
Burrows-Wheeler Transformation

- To create a BWT, the start and end points of the sequence are noted
- Rotations of the given sequence are then constructed by taking the first character of the sequence and placing it at the end of the sequence (step 1).
- Once these sequences are created, they are sorted (step 2).



Effective compression

- From this sorted matrix, the final column is selected as the transformed sequence (step 3).
-
- Different order of transformed sequence bunches repeated bases.
- Also acts as an index.



Want more detail?
Video: https://www.youtube.com/watch?v=zMAa9gFd2Gs (~18-28 min)
Wikipedia: http://en.wikipedia.org/wiki/FM-index

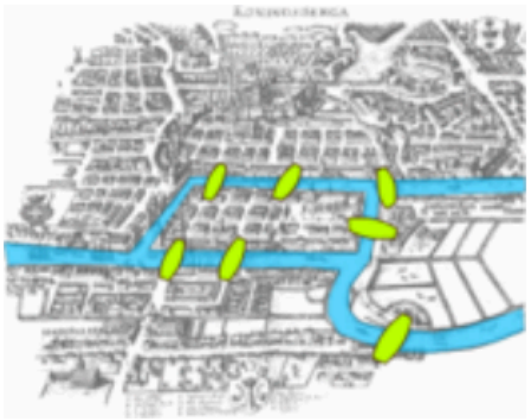
Alignment programs

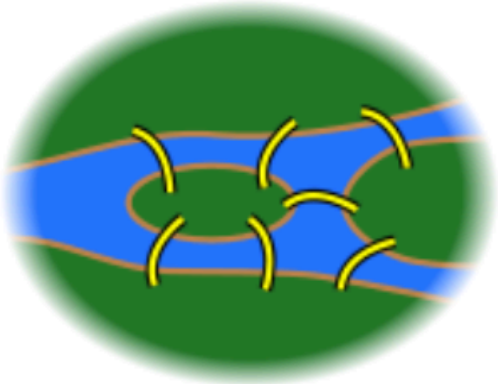
- Lastz (dynamic programming)
- ELAND (Illumina; index reads with hash table)
- MAQ (index reads with hash table)
- NovoAlign (index genome with hash table)
- Bowtie 2 (BWT)
- BWA (BWT)
- BarraCUDA (GPU-accelerated version of BWA)
- TopHat (RNA-Seq)
- Cufflinks (RNA-Seq)
- FANSe (RNA-Seq)

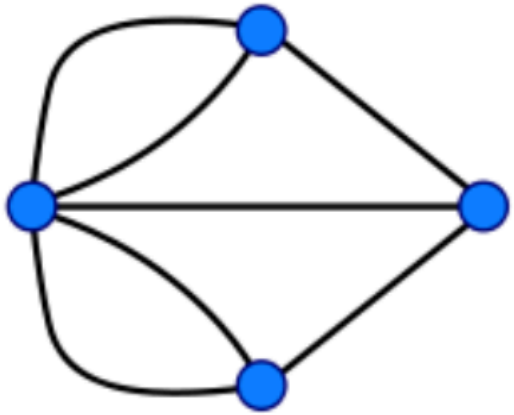
Assembly

- Sequence genome by assembly of short reads
  - Automated dideoxy whole genome shotgun sequencing – reads of ~800bp
  - Reconstruct sequence (assembly)
  - Algorithms work by identifying overlap, then resolving to linear sequence
- Shorter reads/higher coverage from NGS – overlap strategy computationally unfeasible

The Bridges of Königsberg problem



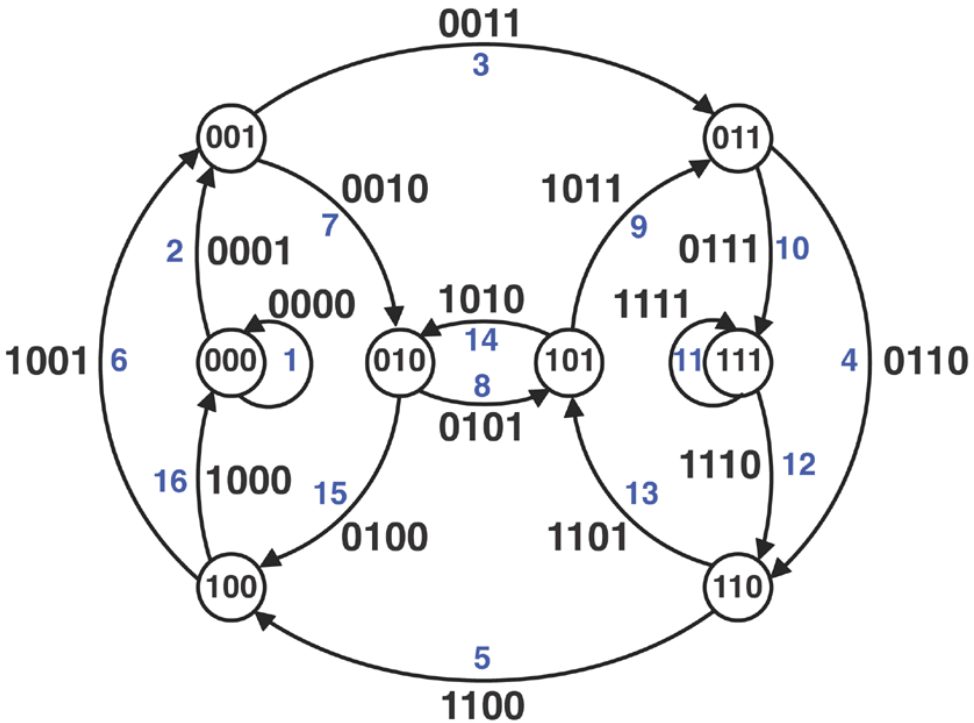




- Königsberg, Prussia (now Kaliningrad, Russia) had seven bridges.
- Can you traverse each bridge exactly once and and return to your starting position? Leonhard Euler solved this in 1735.
- This problem can be represented with edges (bridges) and nodes (land masses) in a graph. Reformulation: is there a Eulerian cycle?
- The answer is no! Except for the start/end node, you have to enter and leave via an edge ⇒ even # of edges required

Euler's theorem and de Bruijn graphs

- Directed graph has Eulerian cycle IFF it is balanced.
- The only place any random (non-repeating) walk ends is at the start!
- Nicolaas de Bruijn (1946) represented kmers as edges connecting nodes (k-1) in length.
- Suffix of one node connects to prefix of another node IFF they are the same = directed edge.
- For 4-mer ACGT:
  - `suffix = ACGT`
  - `prefix = ACGT`
- Eulerian path = smallest possible superstring containing all kmers.

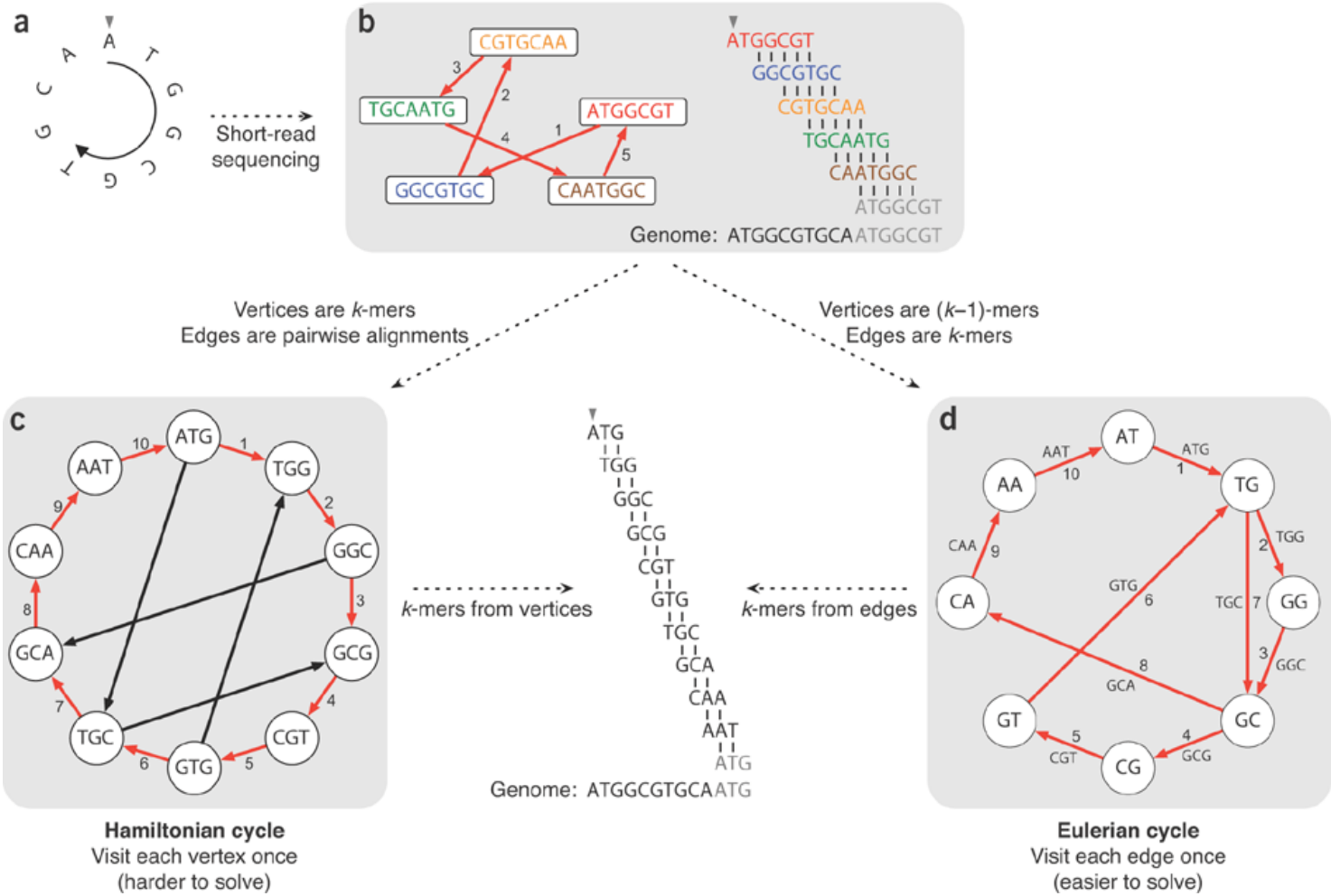


A de Bruijn graph

Hard and easy problems

- How do we find a path that touches each node exactly once? Find the Hamiltonian cycle.
- This problem is hard – strictly NP (non-deterministic polynomial time) hard, meaning:
  - When we've found a solution we can easily verify it, but finding a solution is very difficult.
  - The time taken to find the solution increases exponentially as the number of nodes increases.
- But finding a Eulerian cycle (visiting each edge once) is easy and scales linearly with the size of the graph.

Assembly as Eulerian cycle



Reference: Compeau PE, Pevzner PA, Tesler G. (2011) How to apply de Bruijn graphs to genome assembly. Nat Biotechnol. 29(11):987-91

De Bruijn Assemblers

- Velvet
- ALLPATHS
- ABySS
- SOAPdenovo
- EULER
- Trinity (RNA-Seq)

A larger list of sequence assemblers

Name	Read Type	Algorithm	Reference
SUTTA	long & short	B&B	(Narzisi and Mishra [25], 2010)
ARACHNE	long	OLC	(Batzoglou et al. [14], 2002)
CABOG	long & short	OLC	(Miller et al. [13], 2008)
Celera	long	OLC	(Myers et al. [12], 2000)
Edena	short	OLC	(Hernandez et al. [16], 2008)
Minimus (AMOS)	long	OLC	(Sommer et al. [15], 2007)
Newbler	long	OLC	454/Roche
CAP3	long	Greedy	(Huang and Madan [7], 1999)
PCAP	long	Greedy	(Huang et al. [8], 2003)
Phrap	long	Greedy	(Green [6], 1996)
Phusion	long	Greedy	(Mullikin and Ning [9], 2003)
TIGR	long	Greedy	(Sutton et al. [5], 1995)
ABySS	short	SBH	(Simpson et al. [19], 2009)
ALLPATHS	short	SBH	(Butler et al. [46,47], 2008/2011)
Euler	long	SBH	(Pevzner et al. [17], 2001)
Euler-SR	short	SBH	(Chaisson and Pevzner [35], 2008)
Ray	long & short	SBH	(Boisvert et al. [48], 2010)
SOAPdenovo	short	SBH	(Li et al. [20], 2010)
Velvet	long & short	SBH	(Zerbino and Birney [18,49], 2008/2009)
PE-Assembler	short	Seed-and-Extend	(Ariyaratne and Sung [50], 2011)
QSRA	short	Seed-and-Extend	(Bryant et al. [23], 2009)
SHARCGS	short	Seed-and-Extend	(Dohm et al. [21], 2007)
SHORTY	short	Seed-and-Extend	(Hossain et al. [51], 2009)
SSAKE	short	Seed-and-Extend	(Warren et al. [22], 2007)
Taipan	short	Seed-and-Extend	(Schmidt et al. [24], 2009)
VCAKE	short	Seed-and-Extend	(Jeck et al. [52], 2007)

Reads are defined as "long" if produced by Sanger technology and "short" if produced by Illumina technology . Note that Velvet was designed for micro-reads (e.g. Illumina) but long reads can be given in input as additional data to resolve repeats in a greedy fashion.
doi:10.1371/journal.pone.0019175.t001

Newer
- SGA
- fermi

(use string graph method, BW algorithm to compress and massive parallelisation)

Narzisi G, Mishra B (2011) Comparing De Novo Genome Assembly: The Long and Short of It. PLoS ONE 6(4): e19175. doi:10.1371/journal.pone.0019175
http://www.plosone.org/article/info:doi/10.1371/journal.pone.0019175

PLOS | ONE

Assemblers are stick-shift cars

- With sequence assembly some computational issues arise, e.g., for de Bruijn methods:
  - k-mer multiplicity (caused by repeats in sequence)
  - errors (causing incorrect paths that resemble repeats)
- Key issue is what k-mer size to use (<= read length). Compromise of broken paths/contigs (with large k, if coverage low) versus spurious paths (with low k).
- Paired-end reads should be used to resolve repeat regions and maximum use should be made of multi-threading, multi-core CPUs.
- Some help available, e.g., VelvetOptimizer.pl for testing k, and the VAGUE GUI for point and click.

Competing assemblers

Simulated data

- dnGASP (*de novo* Genome Assembly Project)
- Assemblathon 1

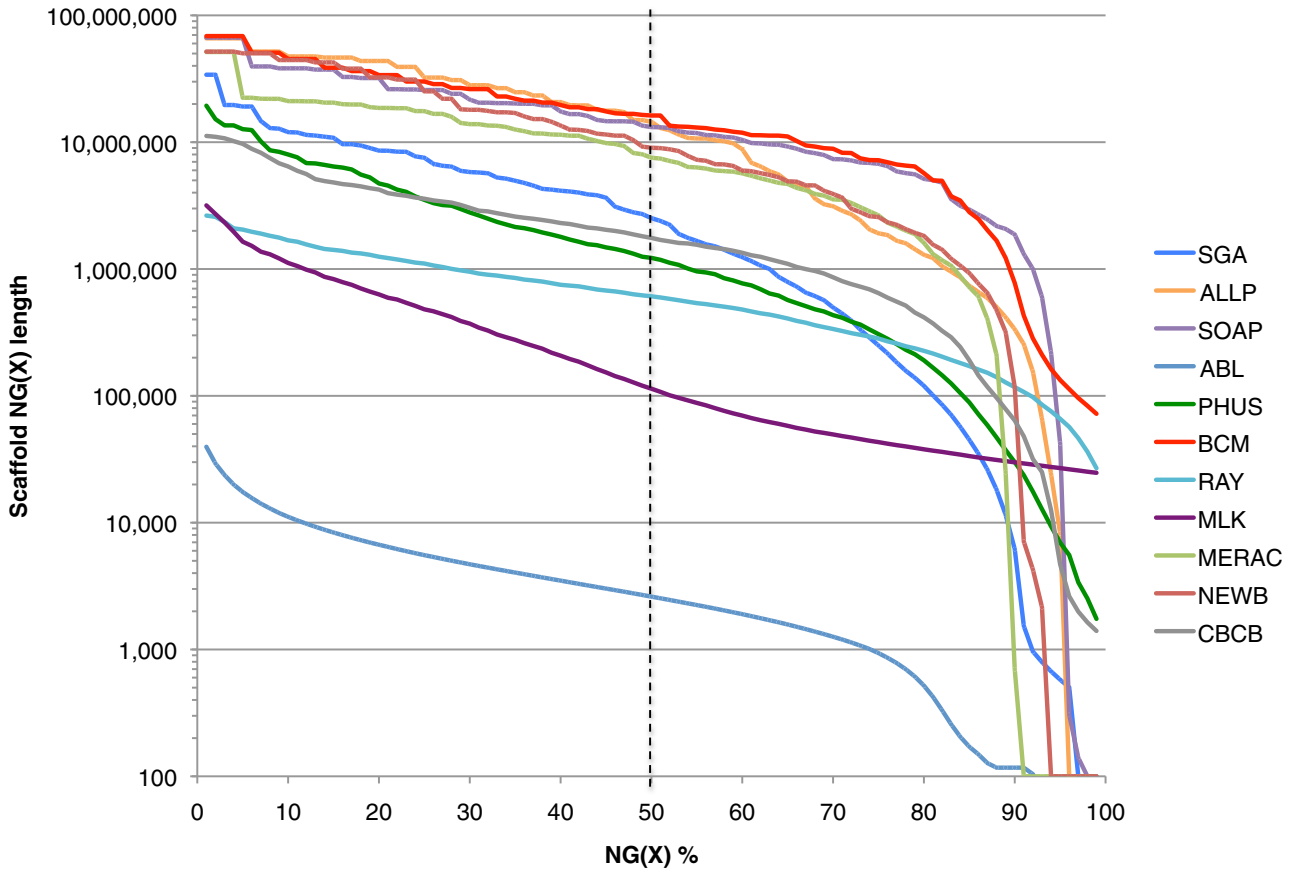
Real data

- GAGE (Genome Assembly Gold-standard Evaluations)
- Assemblathon 2: budgerigar, cichlid fish, boa constrictor – 43 competition entries from 21 teams checked against fosmid sequences

Metrics for contigs: N50 and NG50

- N50 = contig length such that the total length of all contigs this length or longer >=50% total length of contigs.
- So order contigs from longest to shortest:
- 5, 4, 3, 2, 1.
- Total = 15; Half total = 7.5; N50 = 4
- NG50: total = genome size rather than sum of contig lengths.

Budgie NG(X) from Assemblathon 2



READ COMMENTS FOR THIS SLIDE!

Reference: Bradnam KR, et al. (2013) *Assemblathon 2: evaluating de novo methods of genome assembly in three vertebrate species.* Gigascience. 2(1):10

Analysis Platforms

- Sanger Institute (e.g., Dindel for indels) and Broad Institute (GATK pipeline) offer a variety of integrated software for genome analysis.
- Galaxy (http://galaxyproject.org) offers a scalable platform for computational analysis that is point and click:
  - main instance handles mapping, indel identification, ChIP-Seq, GATK pipeline, etc.
  - other instances can be set up in Amazon's cloud.
  - All steps in a workflow are remembered as well as data = reproducible analysis

Full disclosure: I used to work in the Galaxy Project lab at Penn State University, but this is genuinely a popular platform!

Examples: Puerto Rican Parrot

Science bit here: http://dx.doi.org/10.1186/2047-217X-1-14

Panda Genome

- Citation: Li, R., Fan, W., Tian, G., Zhu, H., He, L., Cai, J., ... & Wang, M. (2009). The sequence and de novo assembly of the giant panda genome. *Nature*, *463*(7279), 311-317.

- Illumina GA
- Insert sizes of ~150bp, 500 bp, 2 kb, 5 kb and 10 kb.
- Sequential assembly.
- SOAPdenovo

Learning outcomes

- Grasp strengths and weaknesses of NGS platforms:
  - error rates and sources
- Appreciate future developments:
  - single-molecule sequencing
- Understand key computational concepts behind alignment and assembly:
  - Burrows-Wheeler Transform
  - de Bruijn graphs
  - contigs, scaffolds
  - N50, NG50
- Appreciate future developments:
  - Parallelisation/GPUs