

Introduction to short read NGS:

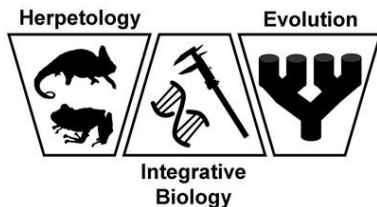
Library construction, UCE capture and ddRADseq

The Natural History Museum, London

Autumn 2021

Instructor: Jeff Streicher

j.streicher@nhm.ac.uk



Litoria iris, Papua New Guinea



Unit 2: Illumina libraries, *de novo* assembly and reference mapping

Lecture



<https://github.com/nhm-herpetology/museum-NGS-training>

Unit 1 review

Lecture

- The Illumina (Solexa) method
- How Illumina sequencing works

Bioinformatics Lab

- How to export Illumina data
- How to clean raw Illumina data

Molecular Lab

- Serapure magnetic beads
- Qubit fluorometry



Unit 2 overview

Lecture

- Inferring larger sequences from short read data
- Read mapping to reference sequences

Bioinformatics Lab

- How to assemble *de novo* contigs from reads
- How to map contigs/reads to references

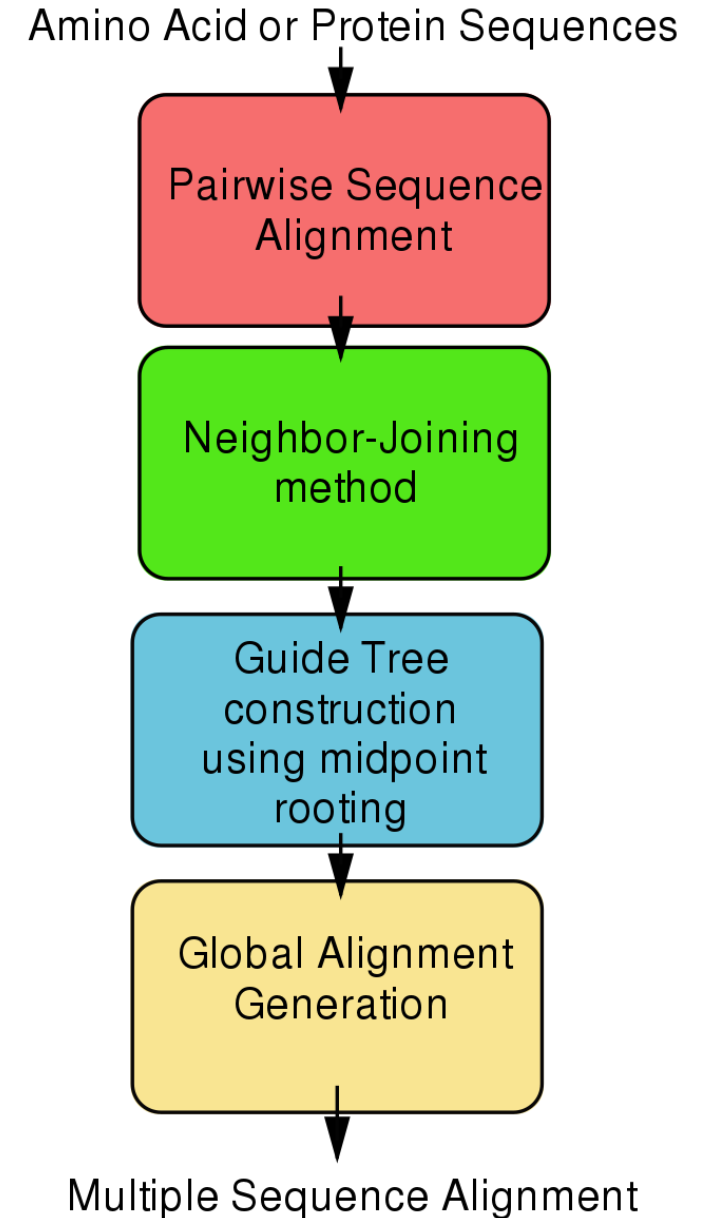
Molecular Lab

- Shotgun Library Prep I (Tomorrow)
- Shotgun Library Prep II (Monday)



DNA sequence alignment

- Multiple sequence alignment
- CLUSTAL (Higgins & Sharp. 1988)
- Started with phylogenetics
- Now used extensively in:
- Genomics
- Physiological research
- Human medicine





CLUSTAL: a package for performing multiple sequence alignment on a microcomputer

Desmond G. Higgins , Paul M. Sharp

[Show more](#) 

[+](#) Add to Mendeley [🔗](#) Share [📄](#) Cite

[https://doi.org/10.1016/0378-1119\(88\)90330-7](https://doi.org/10.1016/0378-1119(88)90330-7)

[Get rights and content](#)

Abstract

An approach for performing multiple alignments of large numbers of amino acid or nucleotide sequences is described. The method is based on first deriving a phylogenetic tree from a matrix of all pairwise sequence similarity scores, obtained using a fast pairwise alignment algorithm. Then the multiple alignment is achieved from a series of pairwise alignments of clusters of sequences, following the order of branching in the tree. The method is sufficiently fast and economical with memory to be easily implemented on a microcomputer, and yet the results obtained are comparable to those from packages requiring mainframe computer facilities.

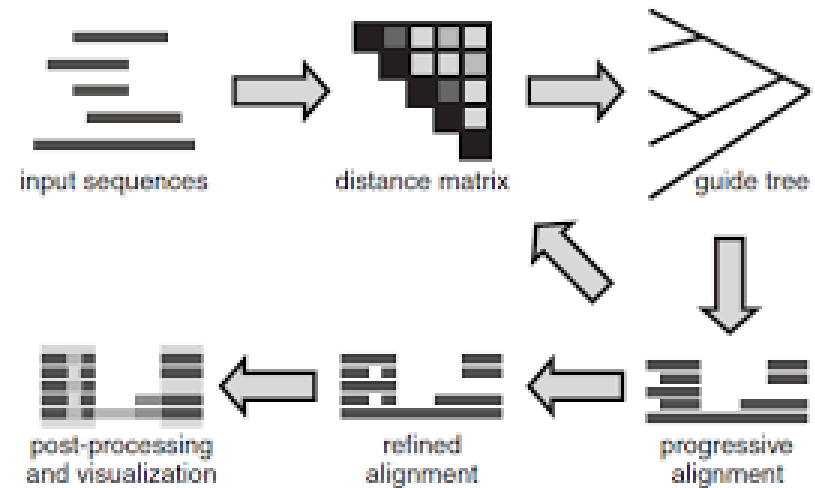


Image by Mónica Chagoyen

⚡

```

Sheep  ---MATSR YEPVAEIGVGAYGTVYKARDPHSGHFVALKSVRVPNGGGAGGGLP ISTVREV 57
Cow    ---MATSR YEPVAEIGVGAYGTVYKARDPHSGHFVALKSVRVPNGGGAGGGLP ISTVREV 57
Human  ---MATSR YEPVAEIGVGAYGTVYKARDPHSGHFVALKSVRVPNGGGAGGGLP ISTVREV 57
Mouse  ---MAATR YEPVAEIGVGAYGTVYKARDPHSGHFVALKSVRVPNGGGAGGGLPVSTVREV 57
Frog    MSKEMKGGYEPVAEIGVGAYGTVYKARDLQSGKFVALKNVRVQTNE--NGLPLSTVREV 57
      :A:AAAAAAAAAAAAAAAAAAAAA:AA:AAAAA:AAA:..:AAA:AAAAAA

Sheep  ALLRRLEAFEPNVRIMDVCAARTDRETKVTLVFEHVDQDLRTYLDKAPPPGLPVE TI 117
Cow    ALLRRLEAFEPNVRIMDVCAARTDRETKVTLVFEHVDQDLRTYLDKAPPPGLPVE TI 117
Human  ALLRRLEAFEPNVRIMDVCAARTDREIKVTLVFEHVDQDLRTYLDKAPPPGLPAETI 117
Mouse  ALLRRLEAFEPNVRIMDVCAARTDRDIKVTLVFEHIDQDLRTYLDKAPPPGLPVE TI 117
Frog    TLLKRL EHFDPNIVKIMDVCAARTDRETKVTLVFEHVDQDLKTYLSKVPPGLPLE TI 117
      :AA:AAA: A:AAA: A:AAAAAA: :AAAA: AAAAAAAAA:AAAA:AAA: A:AAAAAA AAA

Sheep  KDLMRQFLRGLDFLHANCIVHRDLKPE NILVTSGGTVKLADFG LARIYSYQMALTPWVVT 177
Cow    KDLMRQFLRGLDFLHANCIVHRDLKPE NILVTSGGTVKLADFG LARIYSYQMALTPWVVT 177
Human  KDLMRQFLRGLDFLHANCIVHRDLKPE NILVTSGGTVKLADFG LARIYSYQMALTPWVVT 177
Mouse  KDLMRQFLSGLDFLHANCIVHRDLKPE NILVTSNGTVKLADFG LARIYSYQMALTPWVVT 177
Frog    KDLMQFLSGLDFLHLN CIVHRDLKPE NILVTSGGQVKLADFG LARIYSQMALTPWVVT 177
      AAAAA:AAA AA:AAA: AAAAAAAAAAAAAAAAAAAAAA: AAAAAAAAAAAAAAAAAAAAAA

Sheep  LWYRAPEVLLQSTYATFVDMNSVGCI FAEMFRK KPLFCGNS EADQLGKIFDLI GLPPE DD 237
Cow    LWYRAPEVLLQSTYATFVDMNSVGCI FAEMFRK KPLFCGNS EADQLGKIFDLI GLPPE DD 237
Human  LWYRAPEVLLQSTYATFVDMNSVGCI FAEMFRK KPLFCGNS EADQLGKIFDLI GLPPE DD 237
Mouse  LWYRAPEVLLQSTYATFVDMNSVGCI FAEMFRK KPLFCGNS EADQLGKIFDLI GLPPE DD 237
Frog    LWYRAPEVLLQSTYATFVDWNSAGCI FAEMFRK KPLFCGNS EADQLCKIFDII GLPSEE 237
      AAAAAAAAAAAAAAAAAAAAAA:AA: AAAAAAAAA: AAAAAAAAAAAAAAAAAAAAAA AAAA:AAAA: A::

Sheep  WFRDVS LPRGAFSPRGPRFVQSVVPELEESGAQLLLEMLTFNPHKRISAFRALQHSYLHX 297
Cow    WFRDVS LPRGAFSPRGPRFVQSVVPELEESGAQLLLEMLTFNPHKRISAFRALQHSYLHX 297
Human  WFRDVS LPRGAFSPRGPRFVQSVVPEMEESGAQLLLEMLTFNPHKRISAFRALQHSYLHX 297
Mouse  WPREVS LPRGAFSPRGPRFVQSVVPEMEESGAQLLLEMLTFNPHKRISAFRALQHSYLHX 297
Frog    WFDVDT LPRSAFSPRTQQFVDKPVPEIDAMGADLLAMLTFS PQKRIASDAL LHPFFAD 297
      AA: A:AAA:AA:AA: :AA:..AAA:: AA:AAA AAAA: A:AAAAAA AA: A::

Sheep  AE---GDAE----- 303
Cow    AE---GDAE----- 303
Human  DE---GNPE----- 303
Mouse  EE---SDAE----- 303
Frog    DPQACS M QEHFTHICTATDEVK 319

```

‘Assembling’ Illumina data = DNA sequence alignment

‘Assembling’ genomes = DNA sequence alignment

NGS and DNA sequence alignment

- Aligning millions of small (or large) reads is a computationally difficult task
- In the early days of NGS, the available multiple sequence alignment methods were not scalable
- This led to various approaches being developed that can be summarized as either ***de novo* assembly** or **reference alignment**

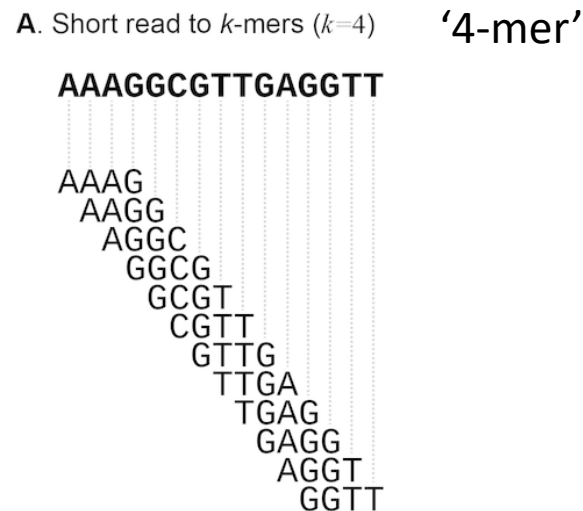


Assembly Terminology

- Reads – short DNA sequences from the Illumina sequencer; these can be single or paired
- Contigs – consensus sequences inferred from alignments of reads
- Scaffolding – linking of non-contiguous DNA sequences with known gap sizes
- K-mer

Assembly Terminology

- Reads – short DNA sequences from the Illumina sequencer; these can be single or paired
- Contigs – consensus sequences inferred from alignments of reads
- Scaffolding – linking of non-contiguous DNA sequences with known gap sizes
- K-mer



De novo assembly

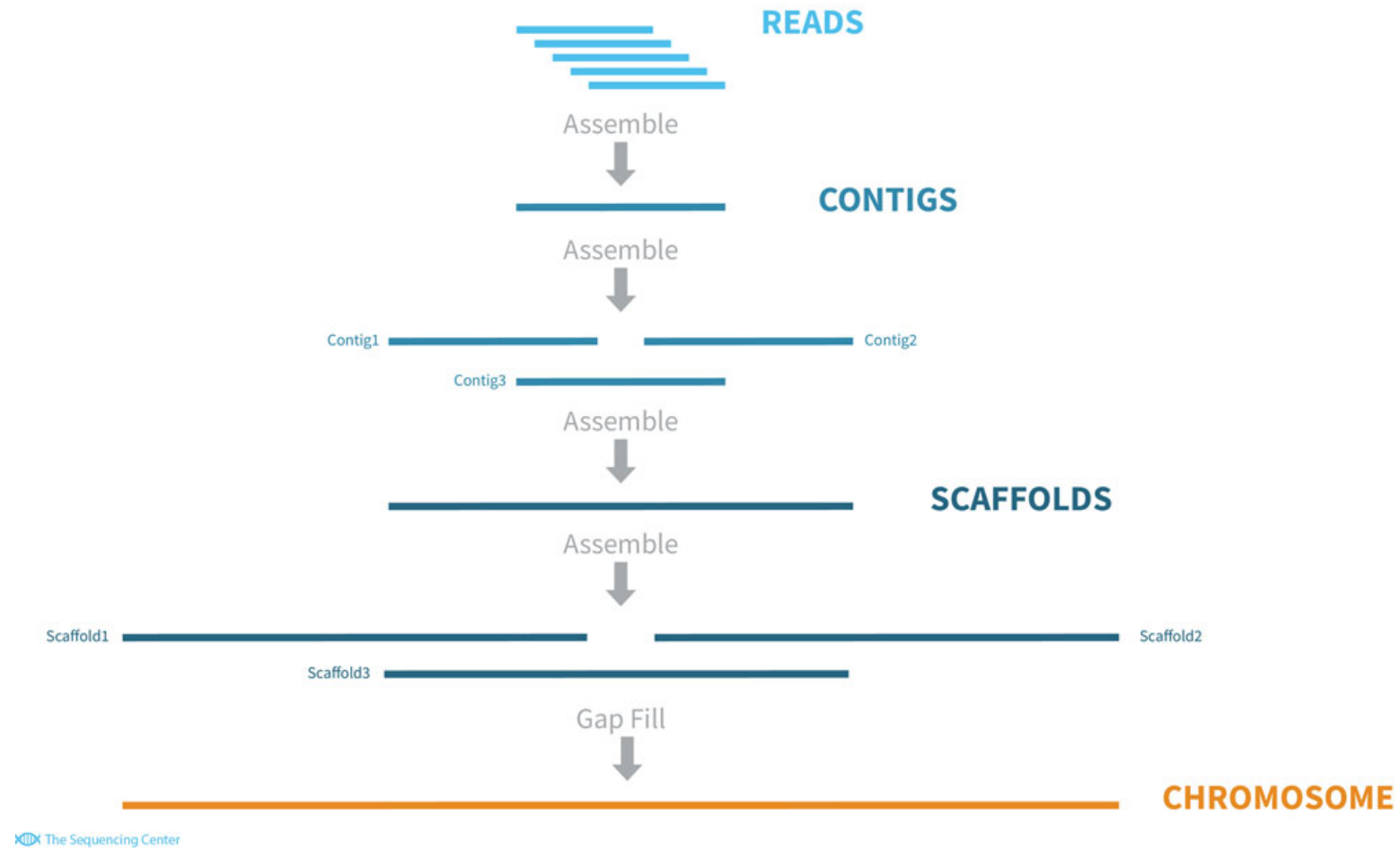


Image from the Sequencing Center

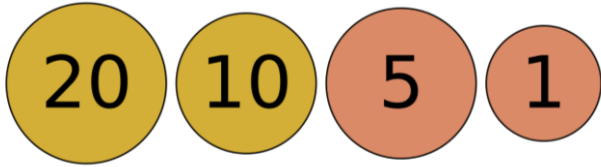
De novo assembly

- Used to infer larger sequences from small reads
- Greedy algorithm assemblers
- Graph method assemblers
- Both assemblers combine reads into larger sequences (i.e. 'contigs')



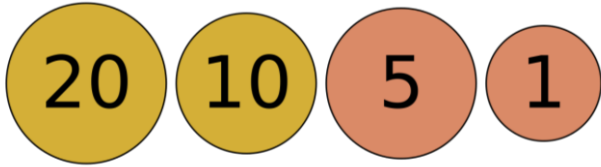
De novo assembly: Greedy algorithms

Change-making problem: Someone has made a purchase and we need to deliver **36 pence** worth of change in an imaginary currency system where there are only four coins:



De novo assembly: Greedy algorithms

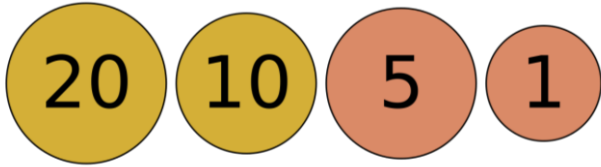
Change-making problem: Someone has made a purchase and we need to deliver **36 pence** worth of change in an imaginary currency system where there are only four coins:




Greedy steps: The algorithm goes through multiple steps until the global optimum is reached in this case being when no more change is owed to the purchaser. In each step, the coin of the highest value, less than the remaining change owed, is the local optimum.

De novo assembly: Greedy algorithms

Change-making problem: Someone has made a purchase and we need to deliver **36 pence** worth of change in an imaginary currency system where there are only four coins:

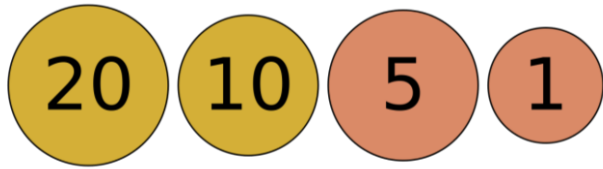


Step 1 $36 - 20 = 16$ 

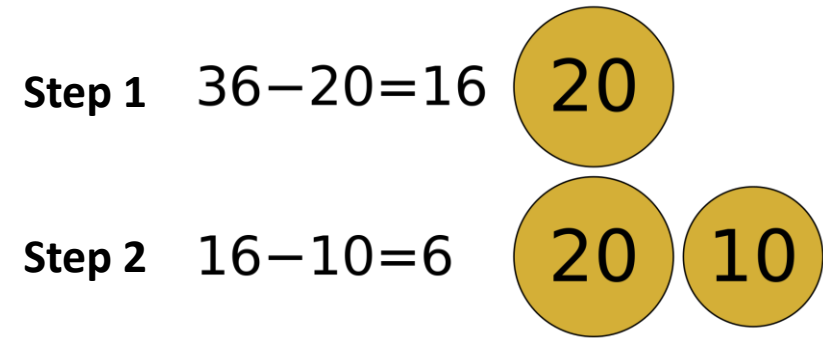
Greedy steps: The algorithm goes through multiple steps until the global optimum is reached in this case being when no more change is owed to the purchaser. In each step, the coin of the highest value, less than the remaining change owed, is the local optimum.

De novo assembly: Greedy algorithms

Change-making problem: Someone has made a purchase and we need to deliver **36 pence** worth of change in an imaginary currency system where there are only four coins:

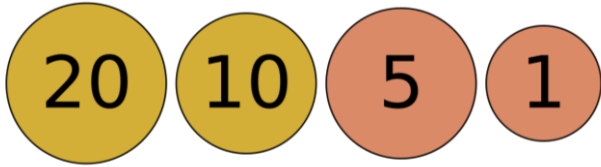


Greedy steps: The algorithm goes through multiple steps until the global optimum is reached in this case being when no more change is owed to the purchaser. In each step, the coin of the highest value, less than the remaining change owed, is the local optimum.

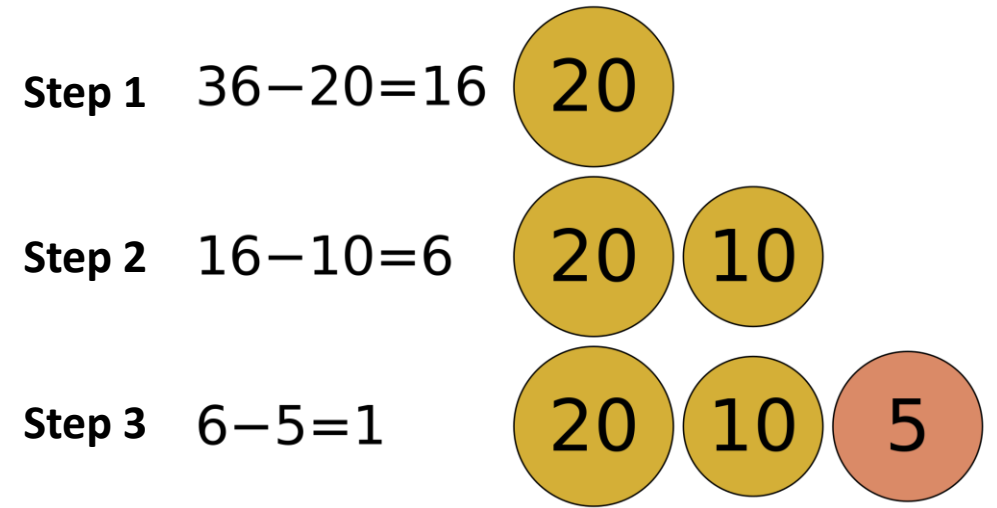


De novo assembly: Greedy algorithms

Change-making problem: Someone has made a purchase and we need to deliver **36 pence** worth of change in an imaginary currency system where there are only four coins:

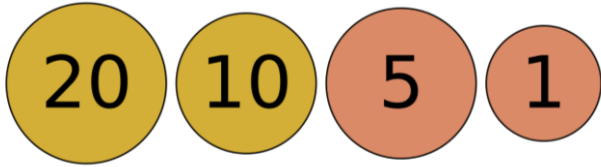


Greedy steps: The algorithm goes through multiple steps until the global optimum is reached in this case being when no more change is owed to the purchaser. In each step, the coin of the highest value, less than the remaining change owed, is the local optimum.

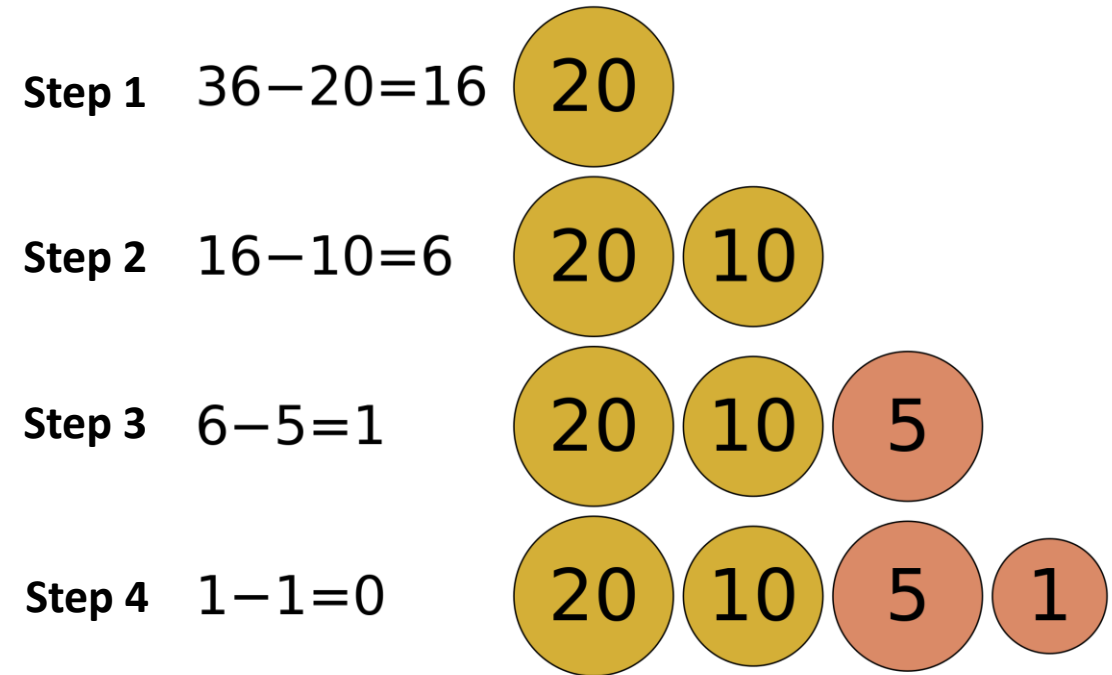


De novo assembly: Greedy algorithms

Change-making problem: Someone has made a purchase and we need to deliver **36 pence** worth of change in an imaginary currency system where there are only four coins:



Greedy steps: The algorithm goes through multiple steps until the global optimum is reached in this case being when no more change is owed to the purchaser. In each step, the coin of the highest value, less than the remaining change owed, is the local optimum.



Global optimum reached in 4 steps!

Greedy algorithms don't always reach the optimal solution...

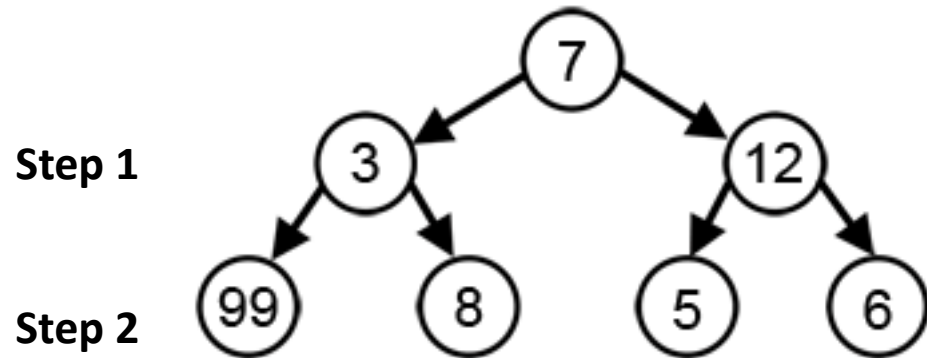
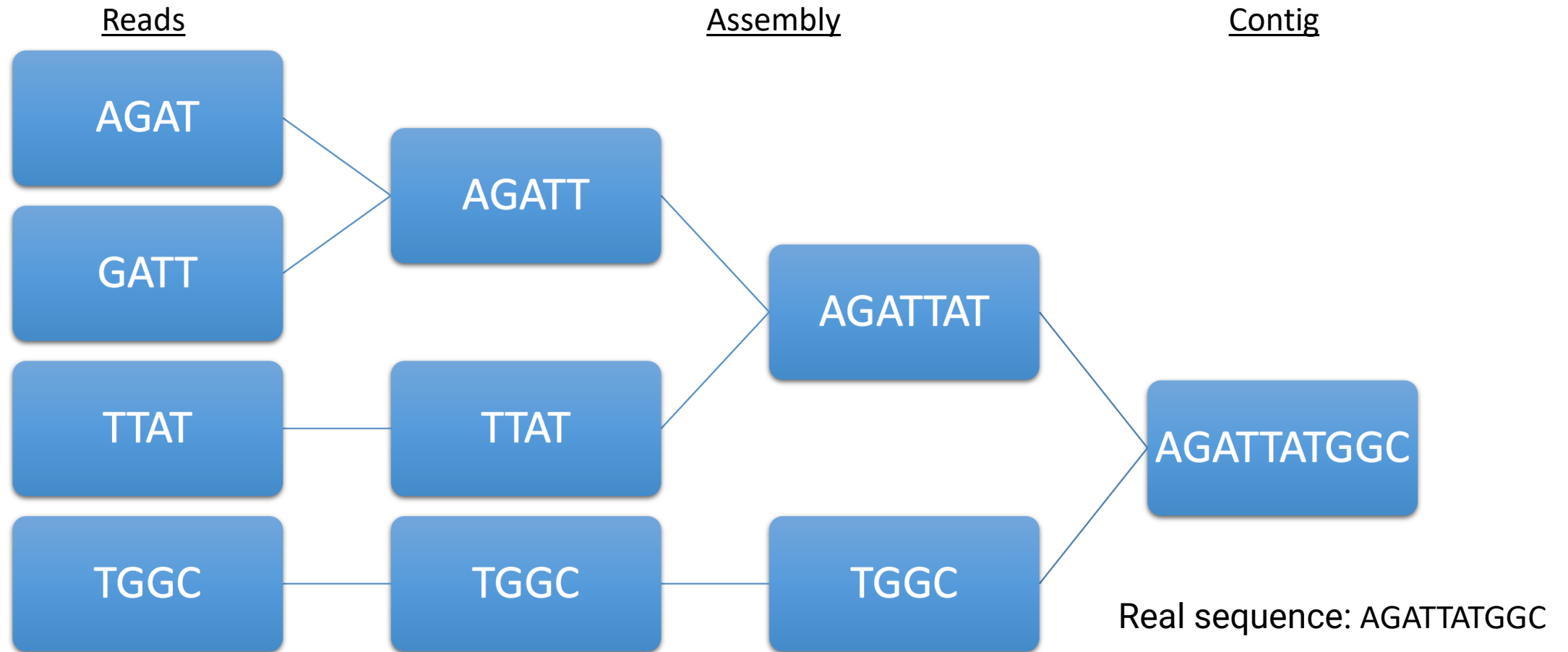


Image by Swfung8 used under a Creative Commons license

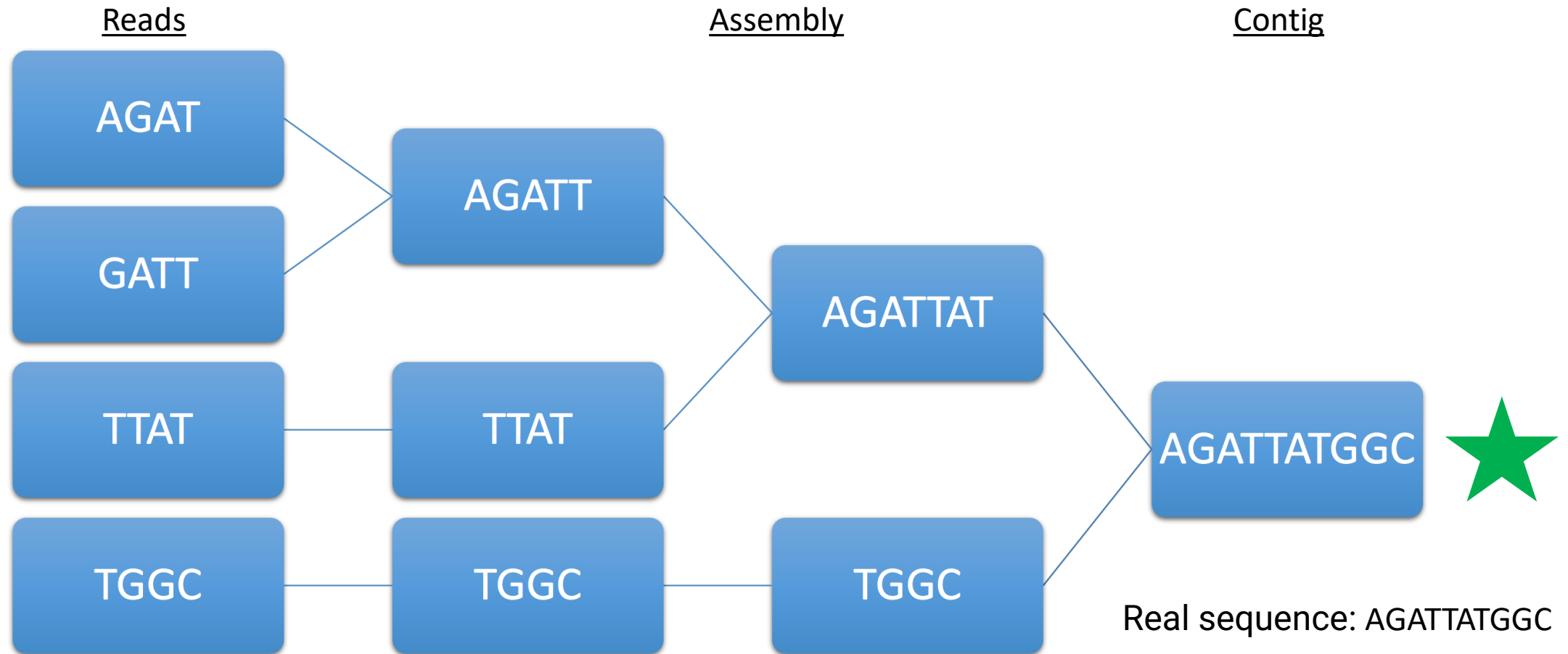
A greedy algorithm is any algorithm that follows the problem-solving heuristic of making the locally optimal choice at each stage. In many problems, a greedy strategy does not produce an optimal solution, **but a greedy heuristic can yield locally optimal solutions that approximate a globally optimal solution in a reasonable amount of time.**

Text *mostly* from Wikipedia ☺

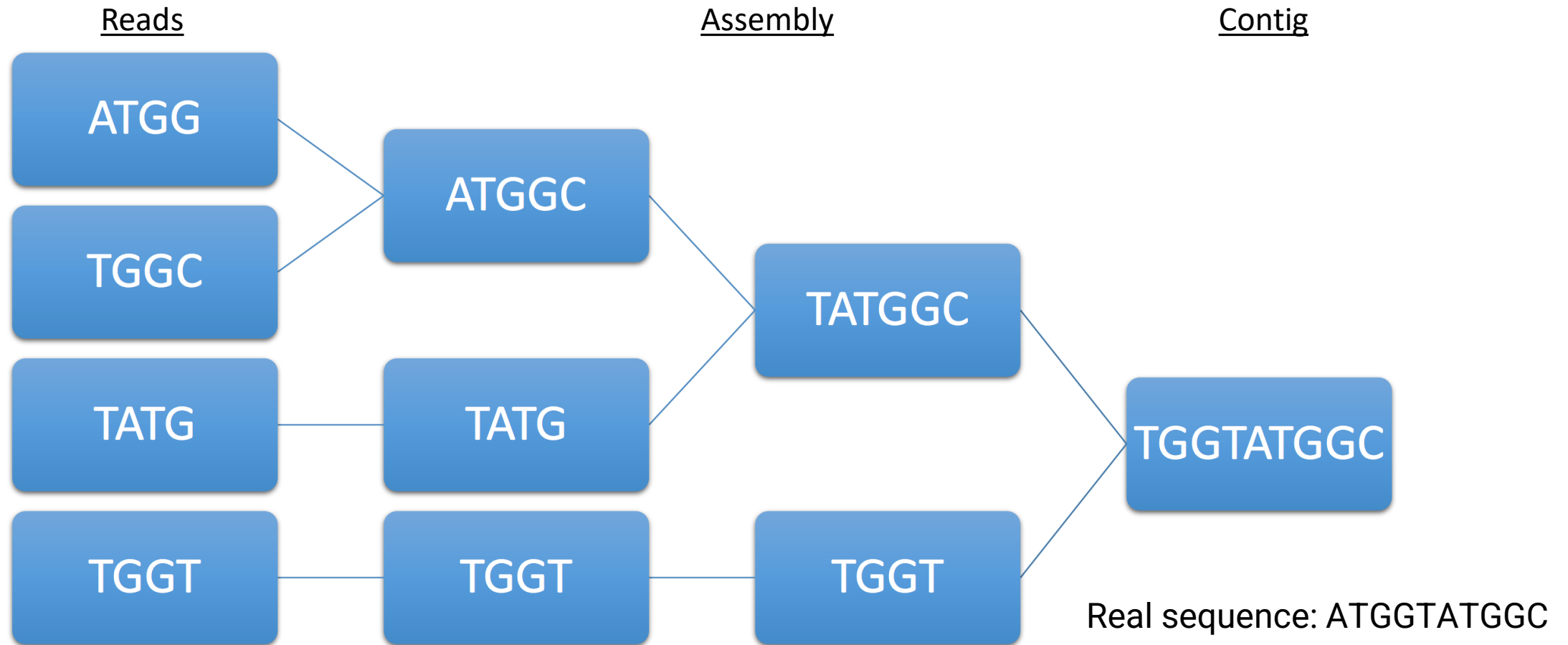
Applying greedy algorithms to short read DNA sequences



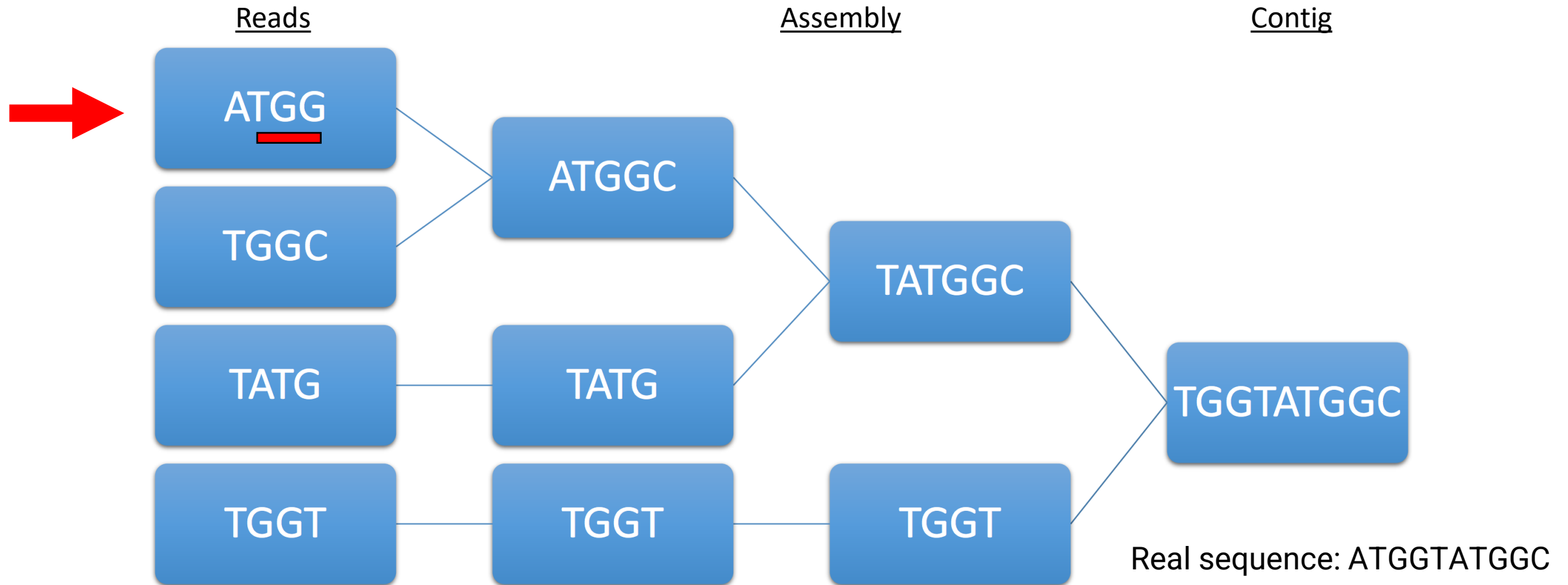
Applying greedy algorithms to short read DNA sequences



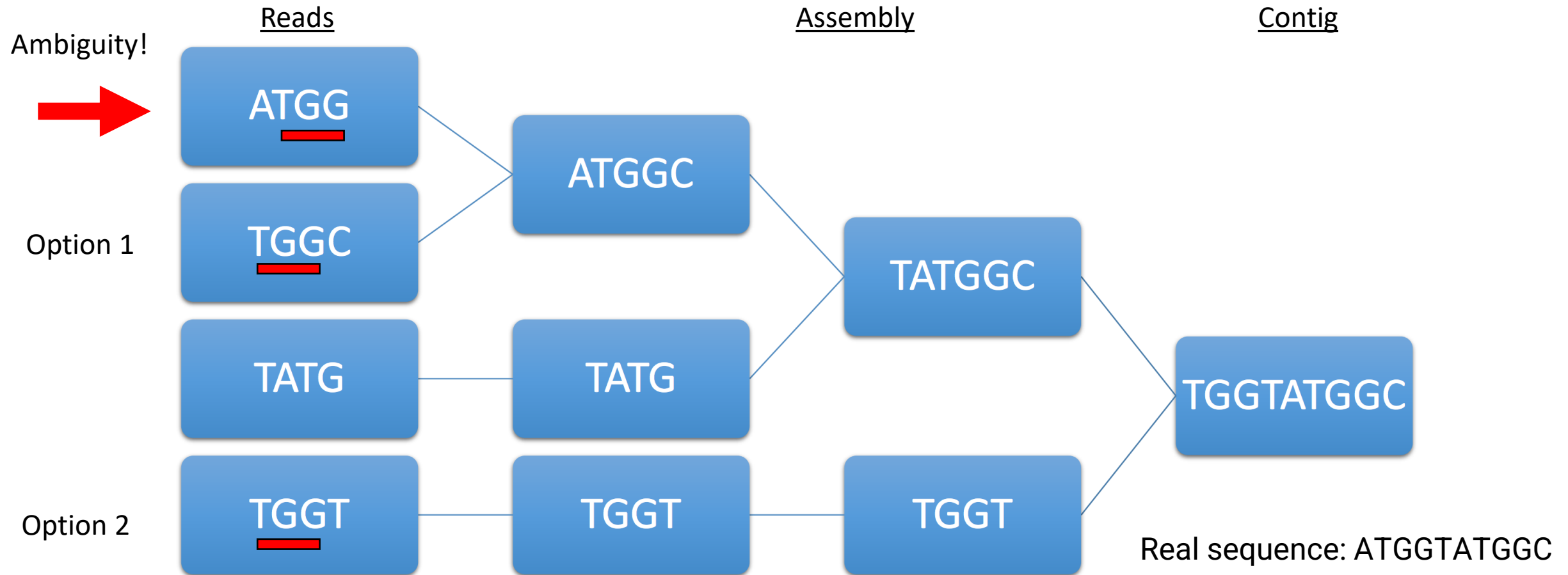
Applying greedy algorithms to short read DNA sequences



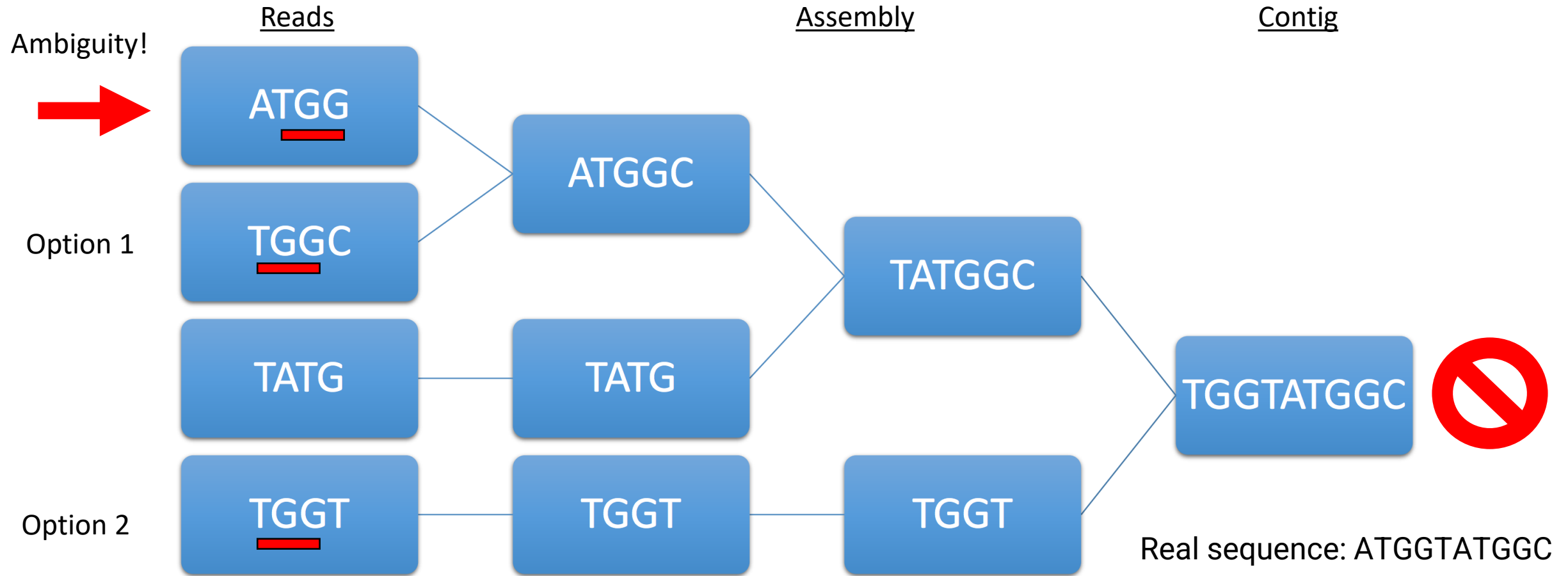
Applying greedy algorithms to short read DNA sequences



Applying greedy algorithms to short read DNA sequences



Applying greedy algorithms to short read DNA sequences



De novo assembly: Graph methods

Two main types:

- String graphs
- **De Bruijn graphs (most assemblers)**

Represent an important step forward in *de novo* assembly because graph methods use algorithms to reach global optima rather than use local optima to estimate the global optimum as we just learned with greedy algorithms.

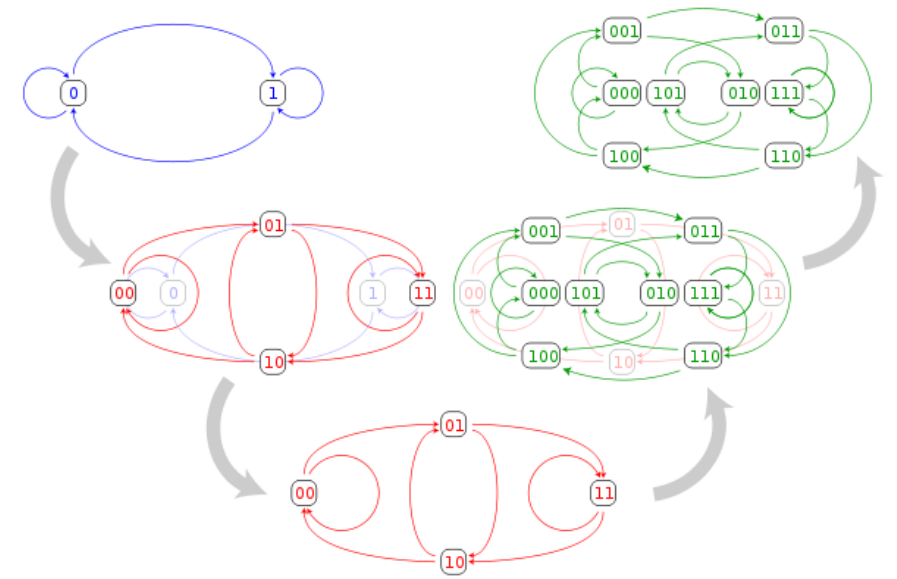


Image by David Eppstein used under a Creative Commons license

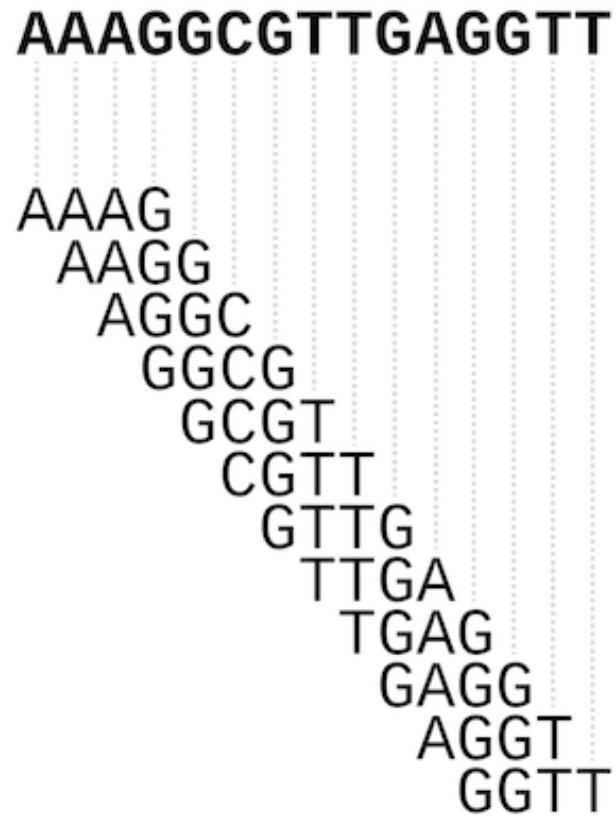
De Bruijn graphs

- Breaks reads into smaller units called k-mers (k)
- Algorithm looks for reads that overlap with k-1 nucleotides
- Each De Bruijn graph has Euler and Hamiltonian cycles

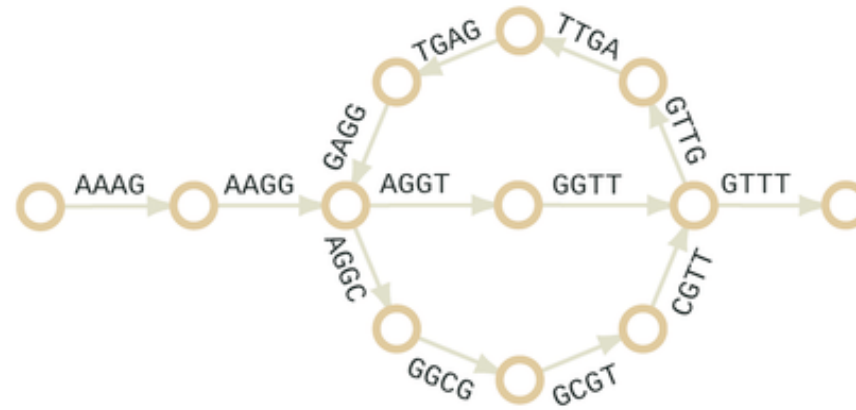


De Bruijn graphs

A. Short read to k -mers ($k=4$)



B. Eulerian de Bruijn graph



C. Hamiltonian de Bruijn graph



De Bruijn graphs

- K-mer size can be specified
- Why use different k-mer sizes?



K-mer size

Informed and automated k -mer size selection for genome assembly ^{FREE}

Rayan Chikhi, Paul Medvedev  [Author Notes](#)

Bioinformatics, Volume 30, Issue 1, 1 January 2014, Pages 31–37,

<https://doi.org/10.1093/bioinformatics/btt310>

Published: 03 June 2013 **Article history** ▼

Table 1. Quality of assemblies for different values of k

Assembly	Contig NG50 (kb)	Size (Mb)	Errors
<i>S.aureus</i> (Velvet)			
$k = 21$	0.5	7.65	0
$k = 31$	19.4	2.83	10
$k = 41$	11.7	2.81	6
$k = 51$	4.6	2.80	9
<i>chr14</i> (Velvet)			
$k = 41$	2.4	74.56	764
$k = 51$	4.0	79.92	843
$k = 61$	5.4	82.10	431
$k = 71$	4.7	81.89	251
$k = 81$	1.8	74.18	153

K-mer size

Informed and automated k -mer size selection for genome assembly ^{FREE}

Rayan Chikhi, Paul Medvedev ✉ [Author Notes](#)

Bioinformatics, Volume 30, Issue 1, 1 January 2014, Pages 31–37,

<https://doi.org/10.1093/bioinformatics/btt310>

Published: 03 June 2013 **Article history** ▼

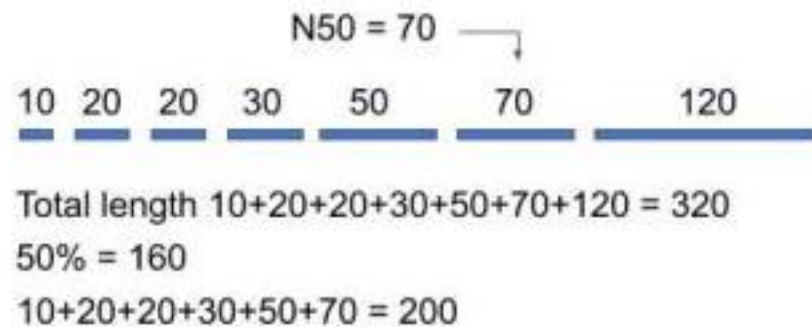
Table 1. Quality of assemblies for different values of k

Assembly	Contig NG50 (kb)	Size (Mb)	Errors
<i>S.aureus</i> (Velvet)			
$k = 21$	0.5	7.65	0
$k = 31$	19.4	2.83	10
$k = 41$	11.7	2.81	6
$k = 51$	4.6	2.80	9
<i>chr14</i> (Velvet)			
$k = 41$	2.4	74.56	764
$k = 51$	4.0	79.92	843
$k = 61$	5.4	82.10	431
$k = 71$	4.7	81.89	251
$k = 81$	1.8	74.18	153

Some assembly summary stats: NG50, N50, L50

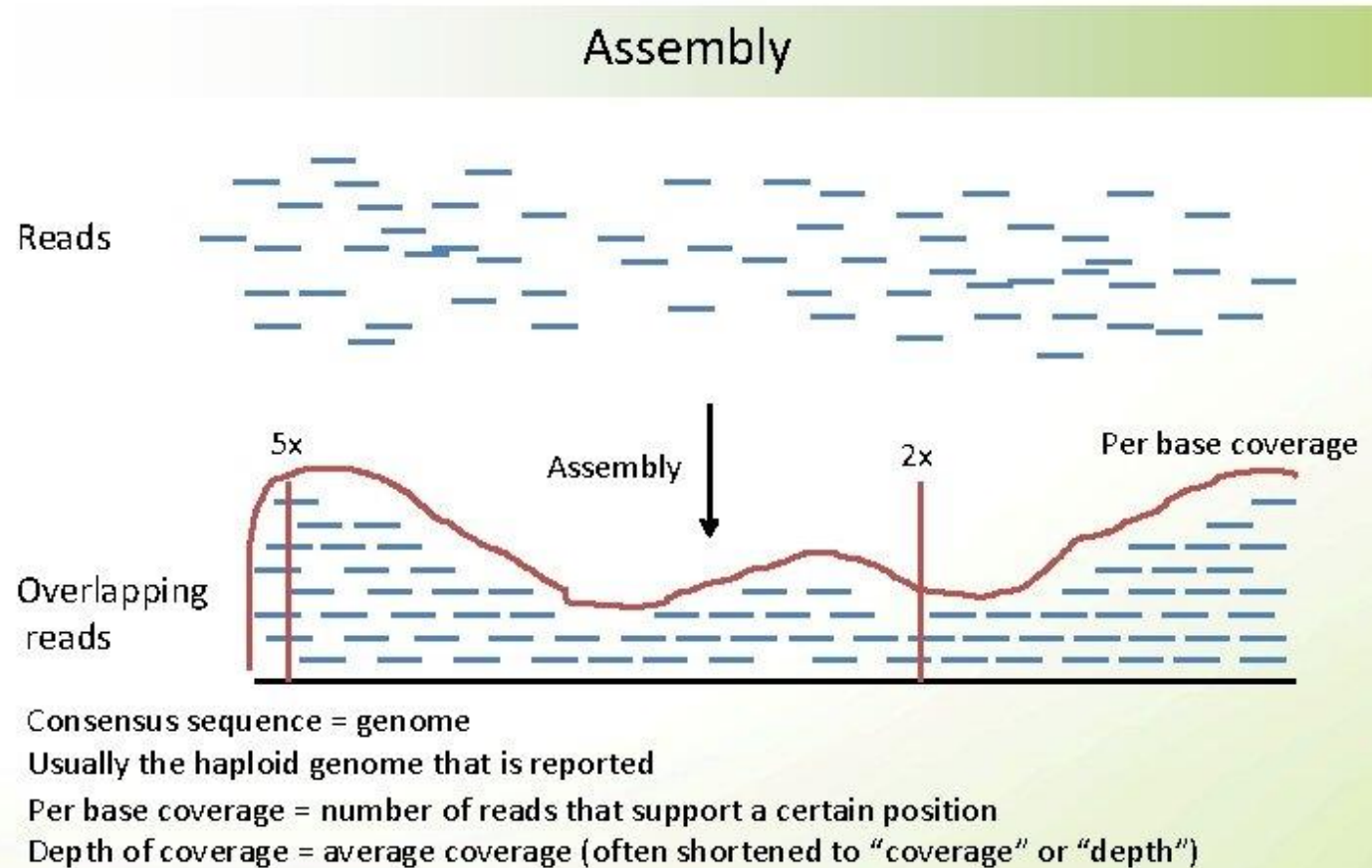
N50

- Summarizes the length of contigs in *de novo* assembly.
- N50 is like a mean or median but it gives greater weight to longer contigs



Example of $N50 = 70$. It means that half (50%) of the assembly includes contigs with at least a length of 70.

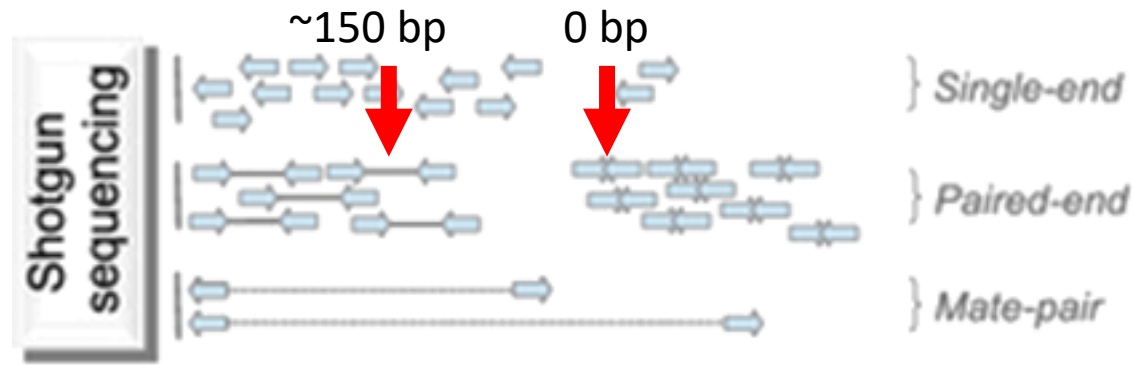
Coverage depth



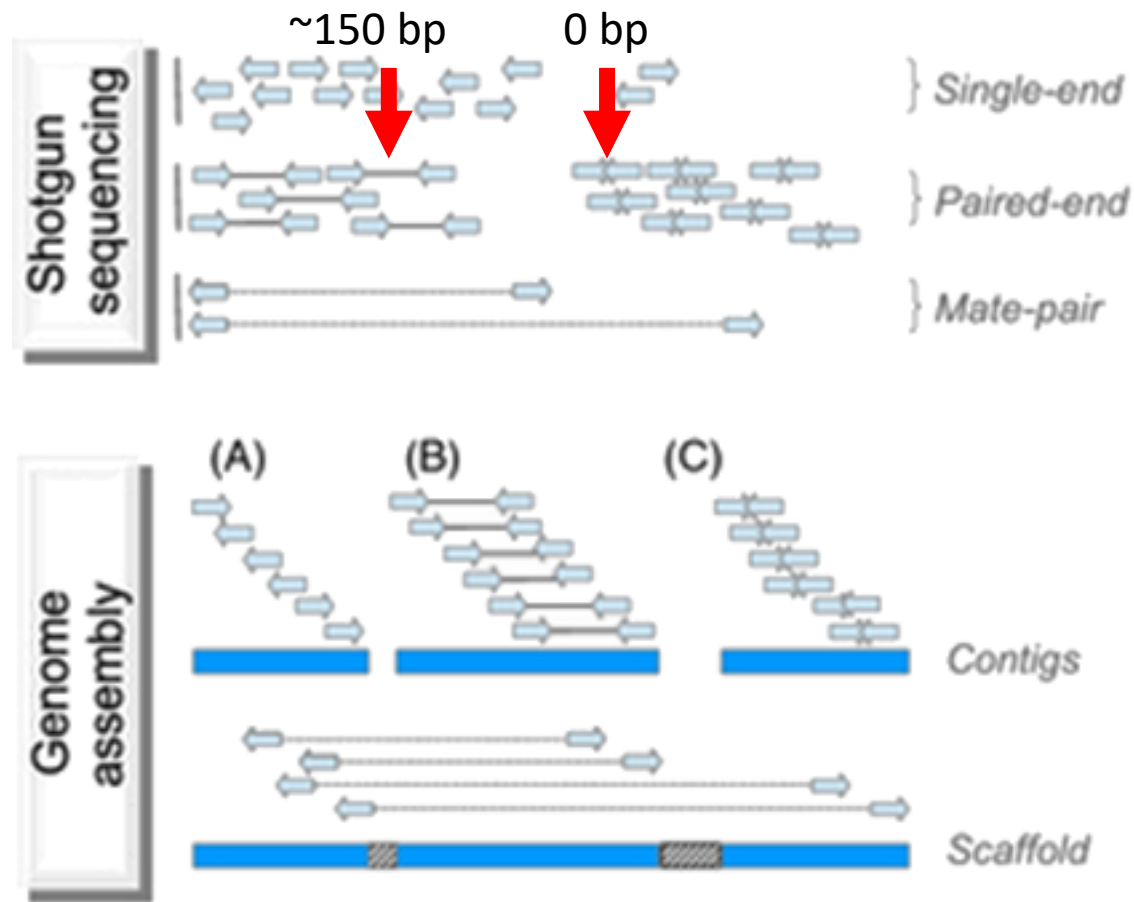
Known insert (or gap) size of paired-end data can help improve assemblies



Known insert (or gap) size of paired-end data can help improve assemblies



Known insert (or gap) size of paired-end data can help improve assemblies



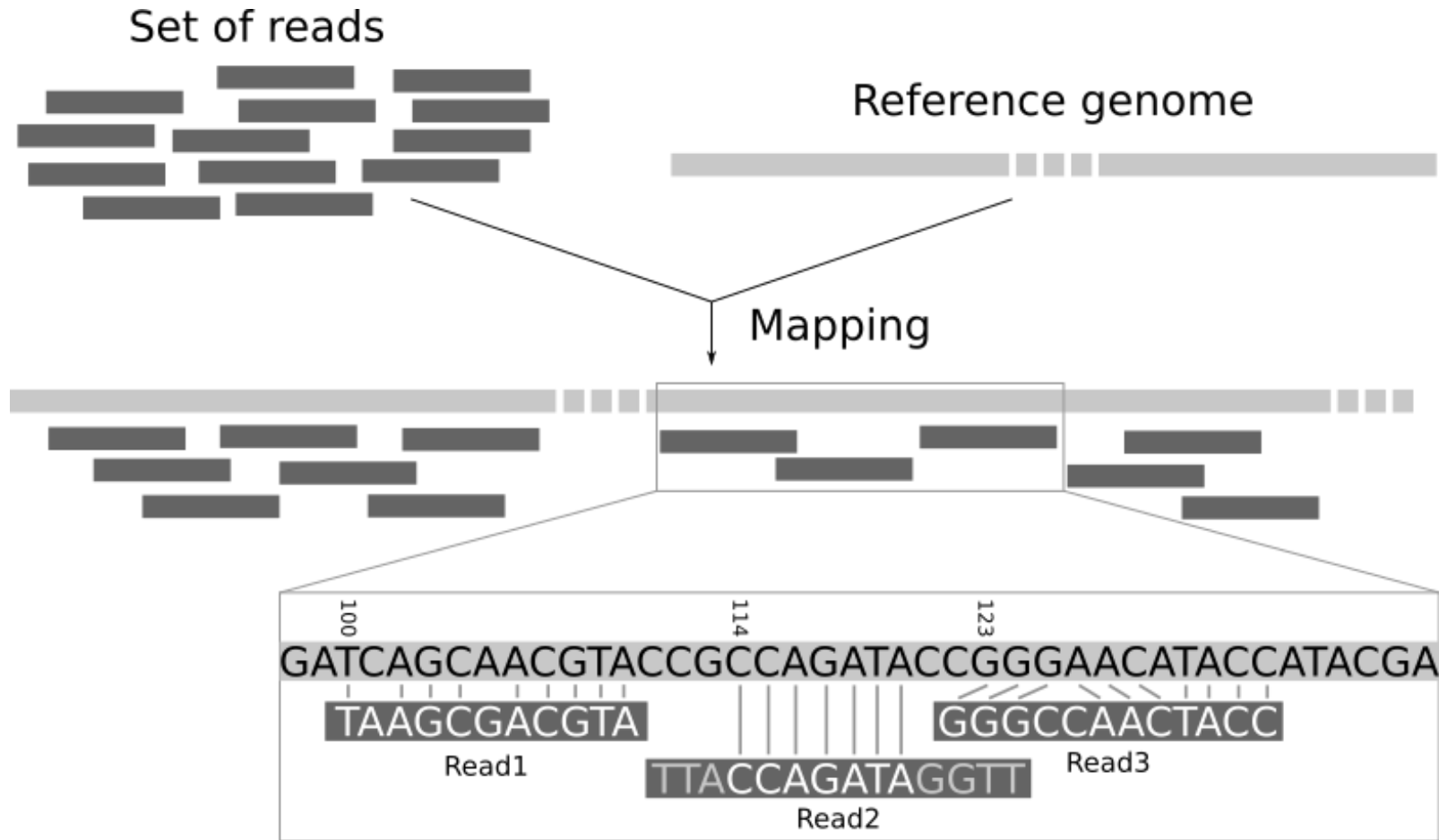
List of de-novo assemblers

Name ↕	Description / Methodology ↕	Technologies ↕	Author ↕	Presented / Last updated ↕	Licence* ↕	Homepage ↕
ABYSS	parallel, paired-end sequence assembler designed for large genome assembly of short reads (genomic and transcriptomic), employ a Bloom filter to De Bruijn graph	Illumina	[8][9]	2009 / 2017	OS	link 🔗
AFEAP cloning Lasergene Genomics Suite	a precise and efficient method for large DNA sequence assembly	two rounds of PCRs followed by ligation of the sticky ends of DNA fragments	[10]	2017 / 2018	C	link 🔗
DISCOVAR	paired-end PCR-free reads (successor of ALLPATHS-LG)	Illumina (MiSeq or HiSeq 2500)	[11]	2014	OS	link 🔗
DNA Baser Sequence Assembler	DNA sequence assembly with automatic end trimming & ambiguity correction. Includes a base caller.	Sanger, Illumina	Heracle BioSoft SRL	2018.09	C (\$69)	NA
DNASTAR Lasergene Genomics	(large) genomes, exomes, transcriptomes, metagenomes, ESTs	Illumina, ABI SOLiD, Roche 454, Ion Torrent, Solexa, Sanger	DNASTAR	2007 / 2016	C	link 🔗
Newbler	genomes, ESTs	454, Sanger	454 Life Sciences	2004/2012	C	link 🔗
Phrap	genomes	Sanger, 454, Solexa	Green, P.	1994 / 2008	C / NC-A	link 🔗
Plass	Protein-level assembler: assembles six-frame-translated sequencing reads into protein sequences	Illumina	[12]	2018 / 2019	OS	link 🔗
Ray	a suite of assemblers including de novo, metagenomic, ontology and taxonomic profiling; uses a De Bruijn graph		[13]	2010	OS	link 🔗
SPAdes	(small) genomes, single-cell	Illumina, Solexa, Sanger, 454, Ion Torrent, PacBio, Oxford Nanopore	[14]	2012 / 2021	OS	link 🔗
Velvet	(small) genomes	Sanger, 454, Solexa, SOLiD	[15]	2007 / 2011	OS	link 🔗
HGAP	Genomes up to 130 MB	PacBio reads	[16]	2011 / 2015	OS	link 🔗
Falcon	Diploid genomes	PacBio reads	[17]	2014 / 2017	OS	link 🔗
Canu	Small and large, haploid/diploid genomes	PacBio/Oxford Nanopore reads	[18]	2001 / 2018	OS	link 🔗
MaSuRCA	Any size, haploid/diploid genomes	Illumina and PacBio/Oxford Nanopore data, legacy 454 and Sanger data	[19]	2011 / 2018	OS	link 🔗
Hinge	Small microbial genomes	PacBio/Oxford Nanopore reads	[20]	2016 / 2018	OS	link 🔗
Trinity	transcriptome assemblies by de Bruijn graph	Illumina RNA-seq	[21]	2011		link 🔗
*Licences: OS = Open Source; C = Commercial; C / NC-A = Commercial but free for non-commercial and academics						

List of de-novo assemblers

Name ⇅	Description / Methodology ⇅	Technologies ⇅	Author ⇅	Presented / Last updated ⇅	Licence* ⇅	Homepage ⇅
ABYSS	parallel, paired-end sequence assembler designed for large genome assembly of short reads (genomic and transcriptomic), employ a Bloom filter to De Bruijn graph	Illumina	[8][9]	2009 / 2017	OS	link
AFEAP cloning Lasergene Genomics Suite	a precise and efficient method for large DNA sequence assembly	two rounds of PCRs followed by ligation of the sticky ends of DNA fragments	[10]	2017 / 2018	C	link
DISCOVAR	paired-end PCR-free reads (successor of ALLPATHS-LG)	Illumina (MiSeq or HiSeq 2500)	[11]	2014	OS	link
DNA Baser Sequence Assembler	DNA sequence assembly with automatic end trimming & ambiguity correction. Includes a base caller.	Sanger, Illumina	Heracle BioSoft SRL	2018.09	C (\$69)	NA
DNASTAR Lasergene Genomics	(large) genomes, exomes, transcriptomes, metagenomes, ESTs	Illumina, ABI SOLiD, Roche 454, Ion Torrent, Solexa, Sanger	DNASTAR	2007 / 2016	C	link
Newbler	genomes, ESTs	454, Sanger	454 Life Sciences	2004/2012	C	link
Phrap	genomes	Sanger, 454, Solexa	Green, P.	1994 / 2008	C / NC-A	link
Plass	Protein-level assembler: assembles six-frame-translated sequencing reads into protein sequences	Illumina	[12]	2018 / 2019	OS	link
Ray	a suite of assemblers including de novo, metagenomic, ontology and taxonomic profiling; uses a De Bruijn graph		[13]	2010	OS	link
SPAdes	(small) genomes, single-cell	Illumina, Solexa, Sanger, 454, Ion Torrent, PacBio, Oxford Nanopore	[14]	2012 / 2021	OS	link
Velvet	(small) genomes	Sanger, 454, Solexa, SOLiD	[15]	2007 / 2011	OS	link
HGAP	Genomes up to 130 MB	PacBio reads	[16]	2011 / 2015	OS	link
Falcon	Diploid genomes	PacBio reads	[17]	2014 / 2017	OS	link
Canu	Small and large, haploid/diploid genomes	PacBio/Oxford Nanopore reads	[18]	2001 / 2018	OS	link
MaSuRCA	Any size, haploid/diploid genomes	Illumina and PacBio/Oxford Nanopore data, legacy 454 and Sanger data	[19]	2011 / 2018	OS	link
Hinge	Small microbial genomes	PacBio/Oxford Nanopore reads	[20]	2016 / 2018	OS	link
Trinity	transcriptome assemblies by de Bruijn graph	Illumina RNA-seq	[21]	2011		link
*Licences: OS = Open Source; C = Commercial; C / NC-A = Commercial but free for non-commercial and academics						

Reference Mapping



Reference mapping

- Maps reads to a reference sequence provided by the user
- The reference sequence is typically a genome (nuclear or mitochondrial), but also could be something small like a mtDNA barcode (e.g. 16S or CO1)



Reference mapping

- Uses many of the same algorithms and alignment strategies of *de novo* assembly
- Differs in that it is only aligning reads to the reference sequence and not reads to each other (as in *de novo* assembly)



Reference mapping

- Longer DNA sequences are inferred by taking the consensus sequence of the mapped reads
- As in *de novo* assembly read coverage can be used to assess confidence in an inferred DNA/genomic sequence



Reference mapping

- Burrows-Wheeler Aligner (BWA)
- Sequence Alignment Map (SAM)
- Binary Alignment Map (BAM)



Burrows-Wheeler transform

- ‘Block-sorting compression’
- Rearranges a character string into runs of similar characters
- Is used for reference mapping because it compresses text in an optimal manner

Input	SIX.MIXED.PIXIES.SIFT.SIXTY.PIXIE.DUST.BOXES
Output	TEXYDST.E.IXIXIXXSSMPPS.B..E.S.EUSFXDIIIOIIIT ^[2]

Burrows-Wheeler transform

- ‘Block-sorting compression’
- Rearranges a character string into runs of similar characters
- Is used for reference mapping because it compresses text in an optimal manner

Input	SIX.MIXED.PIXIES.SIFT.SIXTY.PIXIE.DUST.BOXES
Output	TEXYDST.E.IXIXIXSSMPPS.B..E.S.EUSFXDIIIOIIIT ^[2]



Image from Wikipedia ☺

Fast and accurate short read alignment with Burrows–Wheeler transform

Heng Li, Richard Durbin  [Author Notes](#)

Bioinformatics, Volume 25, Issue 14, 15 July 2009, Pages 1754–1760,
<https://doi.org/10.1093/bioinformatics/btp324>

Published: 18 May 2009 **Article history** ▼

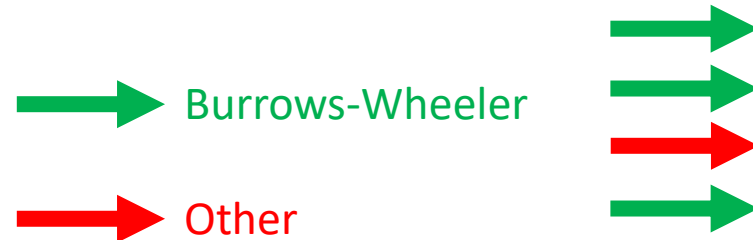


Table 2.

Evaluation on real data

Program	Time (h)	Conf (%)	Paired (%)
Bowtie	5.2	84.4	96.3
BWA	4.0	88.9	98.8
MAQ	94.9	86.1	98.7
SOAP2	3.4	88.3	97.5

The 12.2 million read pairs were mapped to the human genome. CPU time in hours on a single core of a 2.5 GHz Xeon E5420 processor (Time), percent confidently mapped reads (Conf) and percent confident mappings with the mates mapped in the correct orientation and within 300 bp (Paired), are shown in the table.

Fast and accurate short read alignment with Burrows–Wheeler transform

Heng Li, Richard Durbin  [Author Notes](#)

Bioinformatics, Volume 25, Issue 14, 15 July 2009, Pages 1754–1760,

<https://doi.org/10.1093/bioinformatics/btp324>

Published: 18 May 2009 **Article history** ▼

 Burrows-Wheeler

 Other

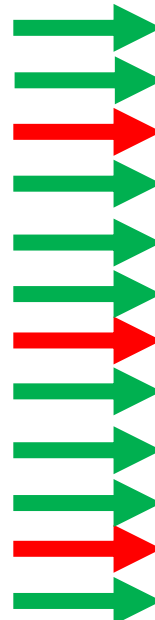


Table 1.


Evaluation on simulated data

Program	Single-end			Paired-end		
	Time (s)	Conf (%)	Err (%)	Time (s)	Conf (%)	Err (%)
Bowtie-32	1271	79.0	0.76	1391	85.7	0.57
BWA-32	823	80.6	0.30	1224	89.6	0.32
MAQ-32	19797	81.0	0.14	21589	87.2	0.07
SOAP2-32	256	78.6	1.16	1909	86.8	0.78
Bowtie-70	1726	86.3	0.20	1580	90.7	0.43
BWA-70	1599	90.7	0.12	1619	96.2	0.11
MAQ-70	17928	91.0	0.13	19046	94.6	0.05
SOAP2-70	317	90.3	0.39	708	94.5	0.34
bowtie-125	1966	88.0	0.07	1701	91.0	0.37
BWA-125	3021	93.0	0.05	3059	97.6	0.04
MAQ-125	17506	92.7	0.08	19388	96.3	0.02
SOAP2-125	555	91.5	0.17	1187	90.8	0.14

One million pairs of 32, 70 and 125 bp reads, respectively, were simulated from the human genome with 0.09% SNP mutation rate, 0.01% indel mutation rate and 2% uniform sequencing base error rate. The insert size of 32 bp reads is drawn from a normal distribution $N(170, 25)$, and of 70 and 125 bp reads from $N(500, 50)$. CPU time in seconds on a single core of a 2.5 GHz Xeon E5420 processor (Time), percent confidently mapped reads (Conf) and percent erroneous alignments out of confident mappings (Err) are shown in the table.

Sequence Alignment Map

The Sequence Alignment/Map format and SAMtools

Heng Li, Bob Handsaker, Alec Wysoker, Tim Fennell, Jue Ruan,
Nils Homer, Gabor Marth, Goncalo Abecasis, Richard Durbin ,
1000 Genome Project Data Processing Subgroup [Author Notes](#)

Bioinformatics, Volume 25, Issue 16, 15 August 2009, Pages 2078–2079,
<https://doi.org/10.1093/bioinformatics/btp352>

Published: 08 June 2009 **Article history** ▼



Col	Field	Type	Brief description
1	QNAME	String	Query template NAME
2	FLAG	Int	bitwise FLAG
3	RNAME	String	References sequence NAME
4	POS	Int	1- based leftmost mapping POSition
5	MAPQ	Int	MAPping Quality
6	CIGAR	String	CIGAR string
7	RNEXT	String	Ref. name of the mate/next read
8	PNEXT	Int	Position of the mate/next read
9	TLEN	Int	observed Template LENgth
10	SEQ	String	segment SEquence
11	QUAL	String	ASCII of Phred-scaled base QUALity+33

Image from Wikipedia 

Binary Alignment Map

- BAM is the compressed version of the SAM format
- Allows for more efficient processing to analyze (e.g. summary stats, visualization etc.) reference maps at large scales



Col	Field	Type	Brief description
1	QNAME	String	Query template NAME
2	FLAG	Int	bitwise FLAG
3	RNAME	String	References sequence NAME
4	POS	Int	1- based leftmost mapping POSition
5	MAPQ	Int	MAPping Quality
6	CIGAR	String	CIGAR string
7	RNEXT	String	Ref. name of the mate/next read
8	PNEXT	Int	Position of the mate/next read
9	TLEN	Int	observed Template LENgth
10	SEQ	String	segment SEquence
11	QUAL	String	ASCII of Phred-scaled base QUALity+33

Image from Wikipedia ☺

Visualizing reference maps

Sources

- Local (0)
- Assembly (4)
 - Concatenated sequences as...
 - DB (1)
 - Normalized Trimmed SRR94...
 - Trimmed SRR949054 (6)
 - Sample Documents (628)
 - Deleted Items (9)
- Shared Databases
- Operations
- Genious Server
- NCBI
- UniProt

Table

Name	Description	HQ%	%GC	Sequence Le...	Post-Trim	# Sequences	Organism	Min Se...
Assembly Report	-	-	-	-	-	-	-	-
Consensus	888,977 reads from Trimmed SRR9490...	99.8%	38.3%	145,673	145,673	-	-	-
Contig	888,977 reads from Trimmed SRR9490...	-	38.3%	156,131	-	888,978	-	14
Unused Reads (Paired)	-	-	35.8%	-	-	20,148	-	10
Unused Reads (Unpaired)	-	-	36.0%	-	-	65,297	-	10
Used Reads	-	-	38.3%	-	-	888,977	-	14

Contig View

Annotations: Lengths Graph, Insert Sizes, Text View, Lineage, Info

Consensus: 1,490

NC_015621

IGGAAGCTTAATCGGAATAAATCAGAATTTTCTCTATGATCGATCGGTATAAACATCAACAGCTACGAATTGGATTAGT

rpoC1 CDS

IGGAAGCTTAATCGGAATAAATCAGAATTTTCTCTATGATCGATCGGTATAAACATCAACAGCTACGAATTGGATTAGT

SRR949054.14066_ILLUMINA-7EDE5D:7:3:13320:20070/1

SRR949054.68941_ILLUMINA-7EDE5D:7:15:10964:17548/2

SRR949054.37782_ILLUMINA-7EDE5D:7:8:7085:20874/2

SRR949054.168987_ILLUMINA-7EDE5D:7:39:1869:14873/2

SRR949054.148122_ILLUMINA-7EDE5D:7:34:9739:9483/2

SRR949054.26152_ILLUMINA-7EDE5D:7:6:17520:11192/1

SRR949054.27153_ILLUMINA-7EDE5D:7:6:8496:15440/1

SRR949054.36412_ILLUMINA-7EDE5D:7:8:2053:15060/2

SRR949054.34674_ILLUMINA-7EDE5D:7:8:15936:7683/2

SRR949054.487811_ILLUMINA-7EDE5D:7:116:3257:12540/2

SRR949054.28729_ILLUMINA-7EDE5D:7:7:17939:2435/2

SRR949054.148880_ILLUMINA-7EDE5D:7:34:1703:14975/2

SRR949054.3115_ILLUMINA-7EDE5D:7:1:11688:14896/2

SRR949054.125896_ILLUMINA-7EDE5D:7:28:1141:18178/1

SRR949054.120199_ILLUMINA-7EDE5D:7:27:18180:11387/2

SRR949054.67530_ILLUMINA-7EDE5D:7:15:12929:11411/1

SRR949054.252967_ILLUMINA-7EDE5D:7:61:7210:9530/1

SRR949054.178459_ILLUMINA-7EDE5D:7:41:11293:20783/1

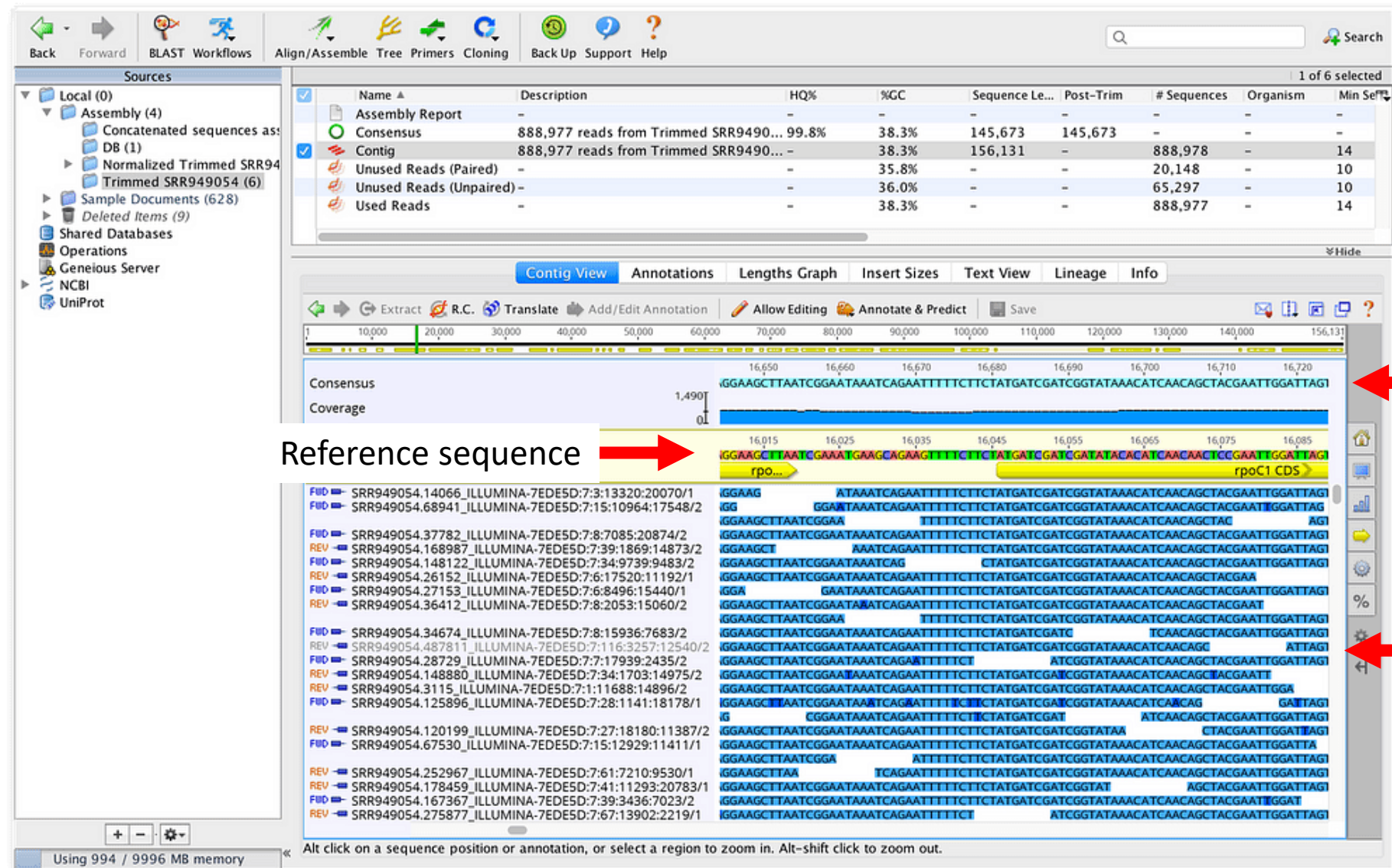
SRR949054.167367_ILLUMINA-7EDE5D:7:39:3436:7023/2

SRR949054.275877_ILLUMINA-7EDE5D:7:67:13902:2219/1

Using 994 / 9996 MB memory

Alt click on a sequence position or annotation, or select a region to zoom in. Alt-shift click to zoom out.

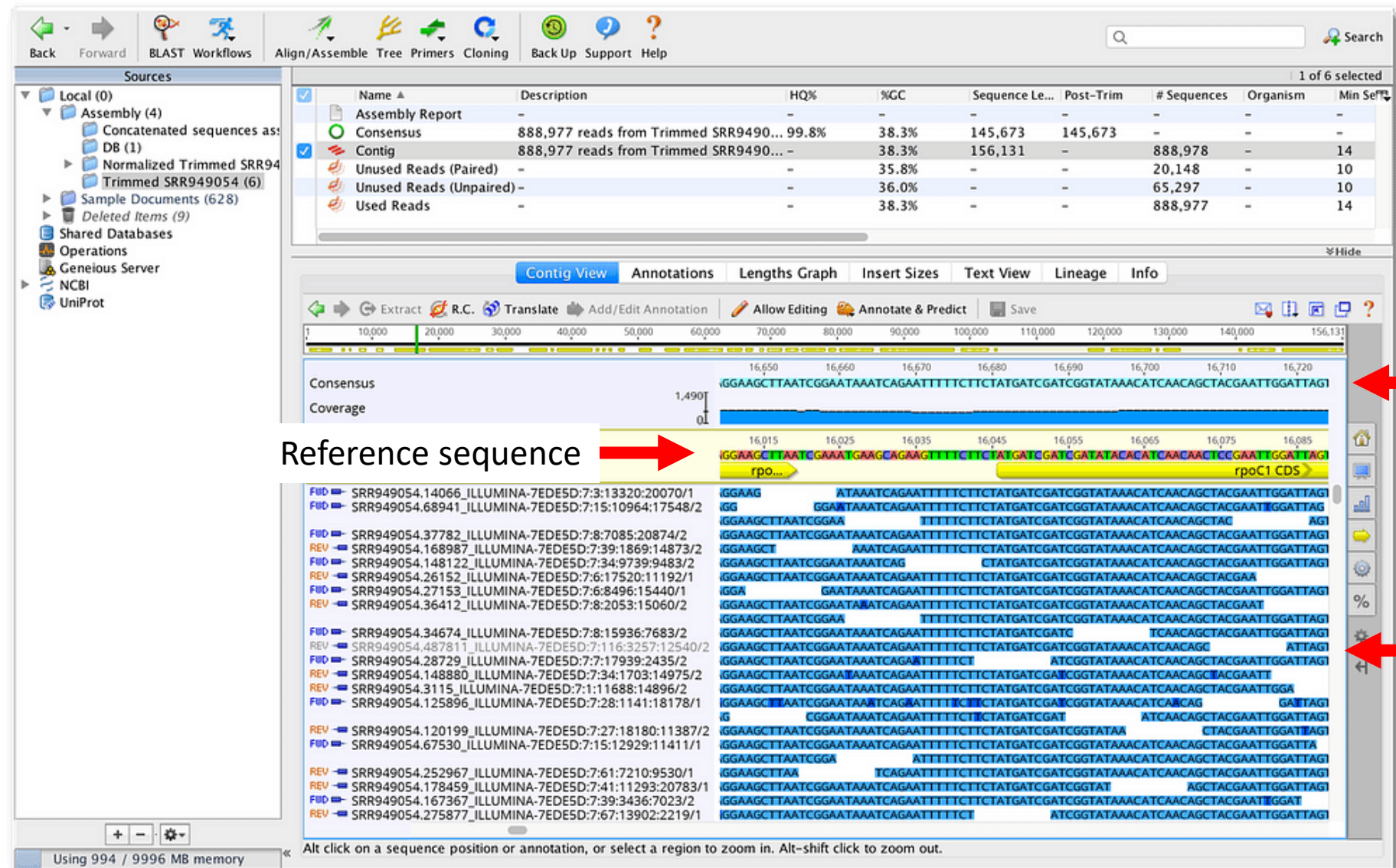
Visualizing reference maps



Consensus of reads

Reads

Visualizing reference maps



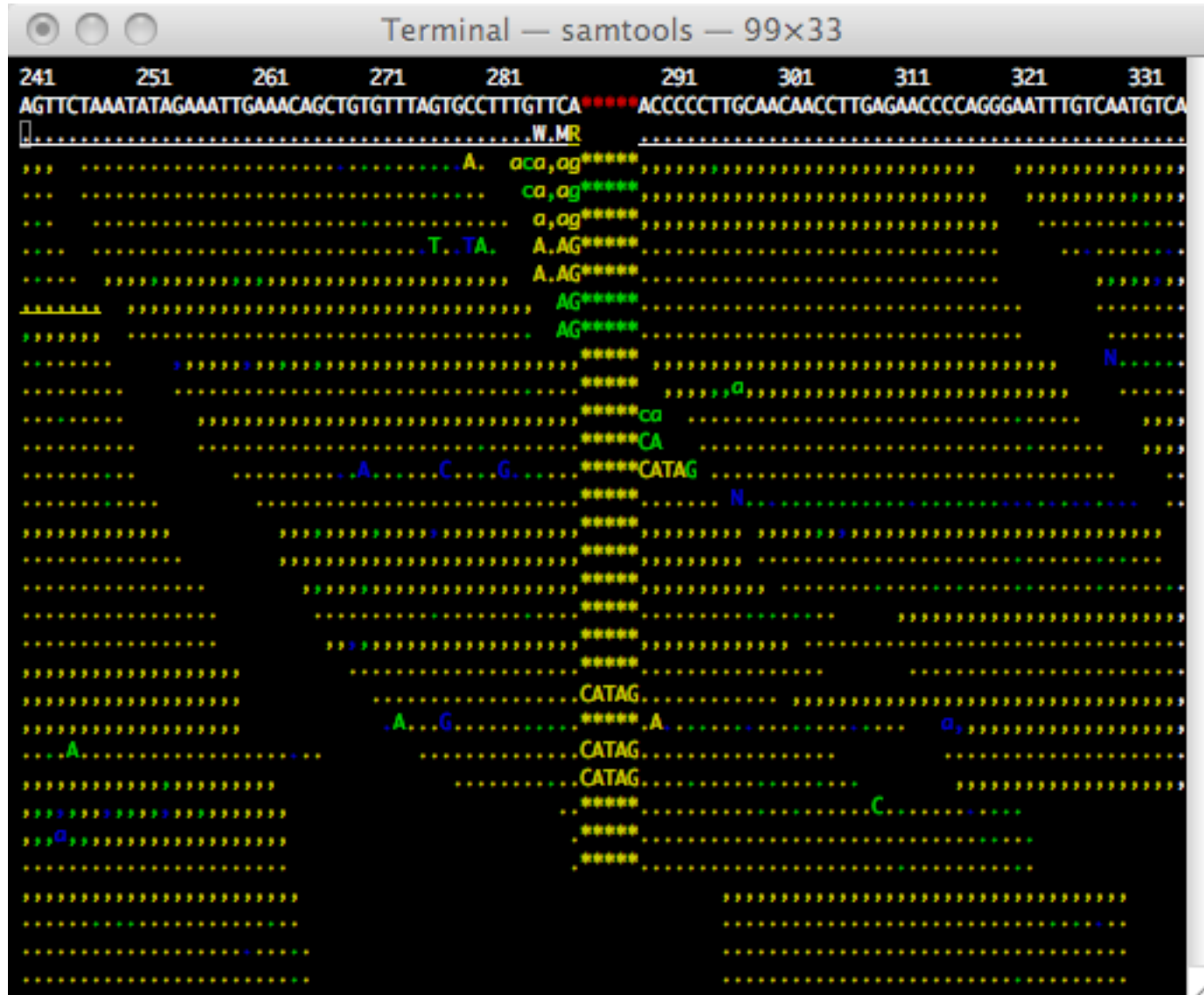
Consensus of reads

Reads

Annual License - £££

geneious
prime

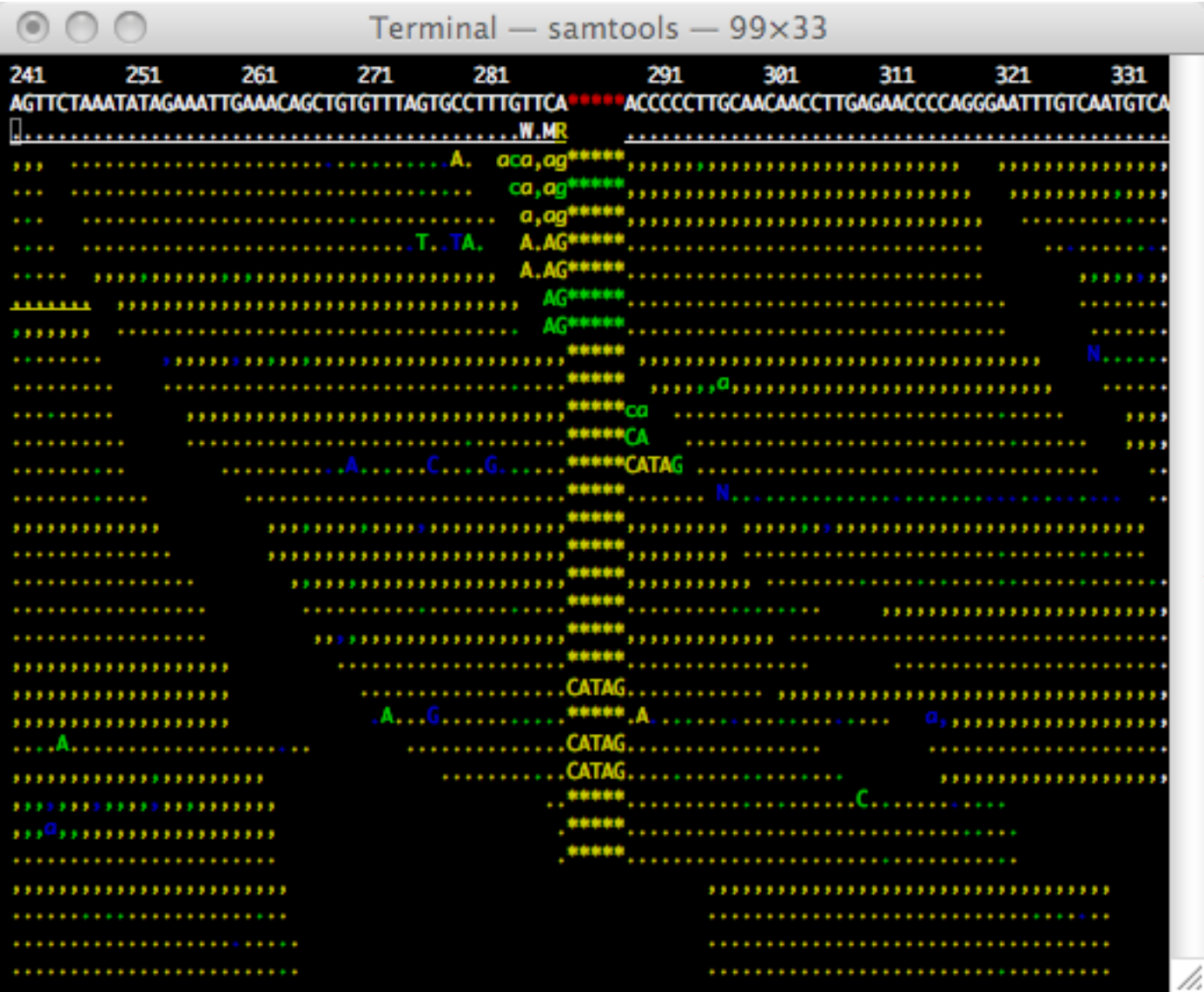
Visualizing reference maps



SAMtools

Visualizing reference maps

Reference sequence →



← Consensus of reference/reads

← Reads

SAMtools

Reference bias



Image from Seven Bridges Genomics

Reference bias



Reference bias

Insertion not in the Reference Sequence
or
Deletion not in the Reads



Reads →



Reference sequence →

LINEAR REFERENCE

Summary

- What method is best for your Illumina data?
- Depends on the application...



Summary

- Whole genome sequencing (*de novo* assembly + reference mapping)
- Sequencing of hundreds of loci like UCEs (*de novo* assembly)
- Gene expression analysis with transcriptomes (reference mapping)



Unit 2: Illumina libraries, *de novo* assembly and reference mapping

Bioinformatics Lab



<https://github.com/nhm-herpetology/museum-NGS-training>

Getting ready

- Navigate to the 'NGS_course' directory we made last week
- mkdir Unit_2