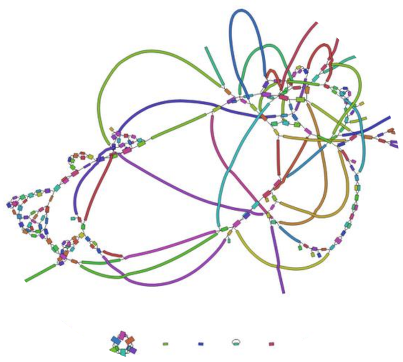
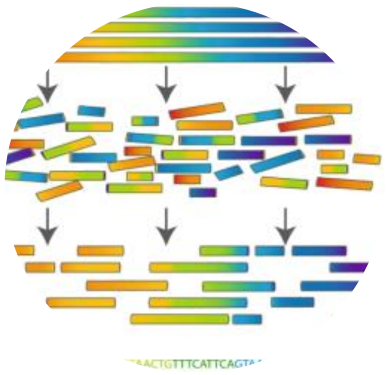
PMB2025

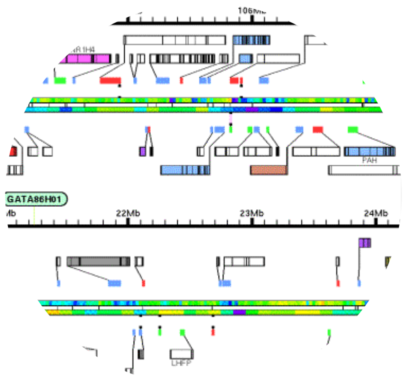
PATHOGEN MULTIOMICS AND BIOINFORMATICS

Recife PE 2025

Module 2: *De novo* Assembly





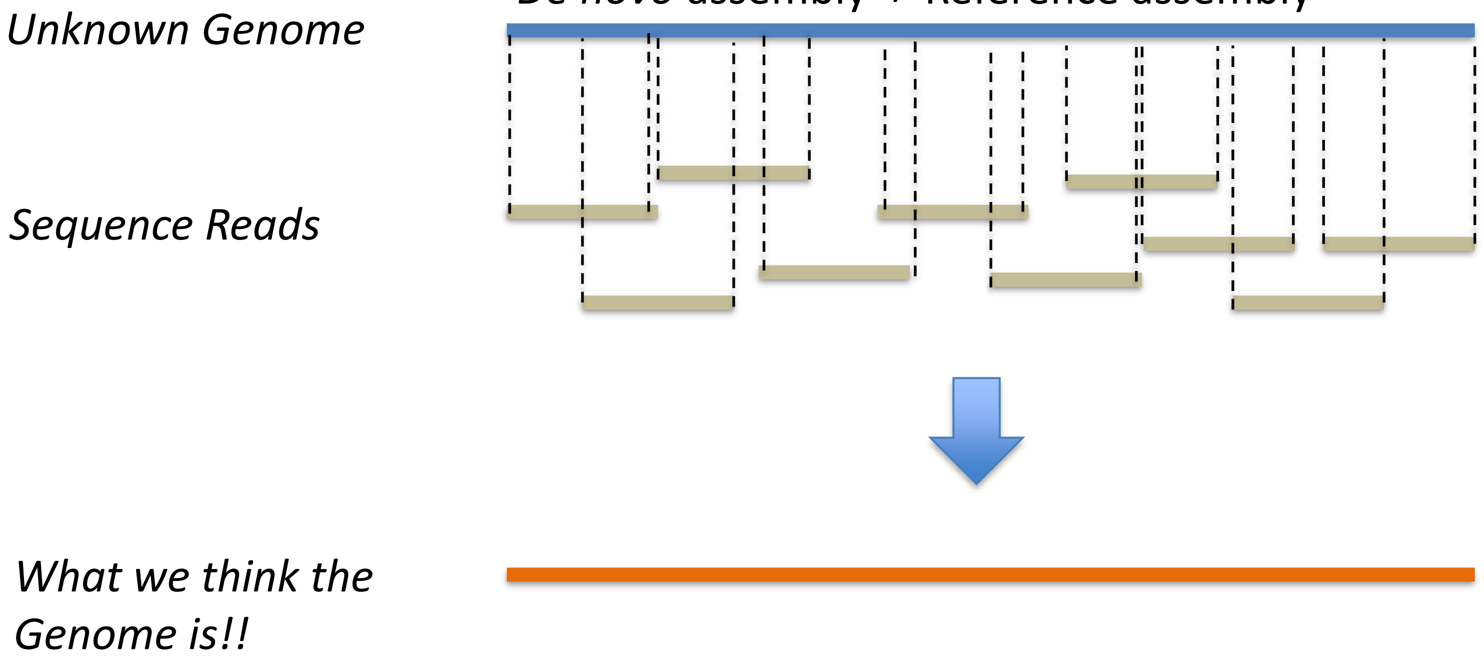


João Perdigão

De novo Assembly

De novo assembly – genome assembly without a reference genome, i.e., starting *de novo* from sequence data

De novo assembly ≠ Reference assembly



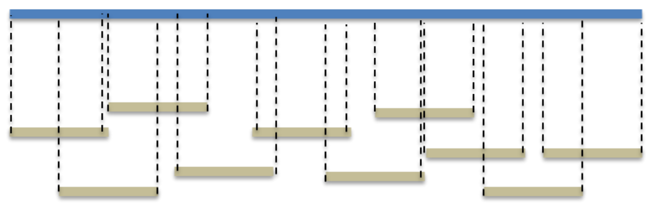
A contig represents a stretch of DNA sequence assembled without any gaps, where the order and orientation of the nucleotides are known with high confidence.

De novo Assembly

*de novo a*ssembly attempts to reconstruct genomes by exploring read overlapping and contiguity

Problems and Challenges:

- Large volume of sequence reads
- Sequencing errors
- Genomic repeat patterns/regions and homopolymers
- Uneven coverage/sequencing



But, what is the definition of an assembly?

Best set of sequences that can approximate the sequenced genetic material

Implications?

imed Research Institute for Medicines

Why to assemble?

Objective/Purpose of the Assembly:

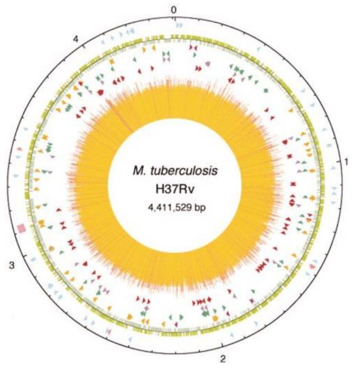
Obtain a reference genome

↓

PacBio

↓

Finished Genome Assembly



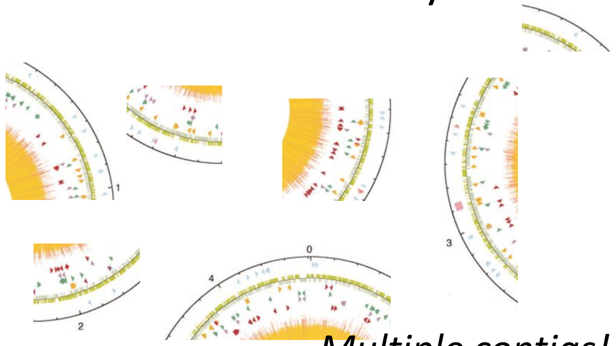
Gene content, insertions, deletions

↓

Illumina

↓

Draft Genome Assembly



Multiple contigs!

←

Manual closure
Mate-pair sequencing

Assembly Methods

Three major methods for assembly:

i) overlap-and-extend

Finds read overlap and extends – suffix of a read is equal to the prefix of another read with a length that meets a defined threshold.

Software: SSAKE, VCAKE and SHARCGS

ii) string graph

Construction of string graph where each read is a vertex with edges connecting overlapping nodes.

Software: Edena and BOA

Problems: high memory comsumption and sequencing errors

Assembly Methods

Three major methods for assembly:

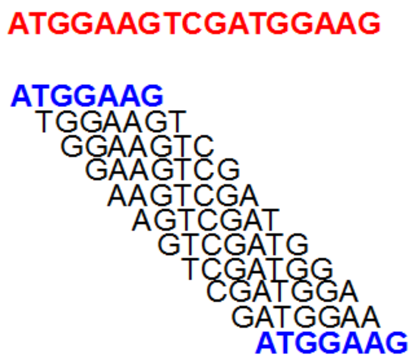
iii) de Bruijn Graph

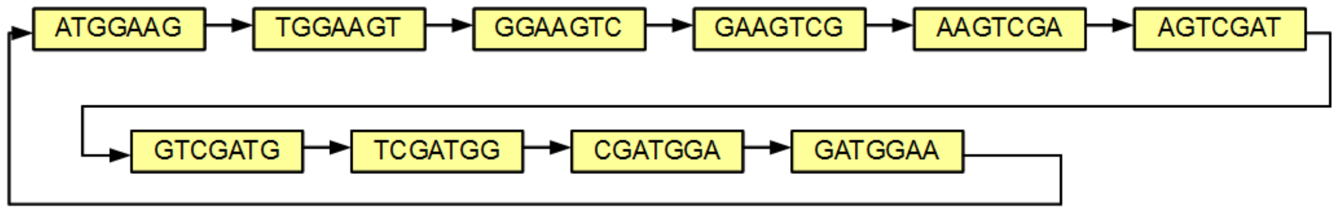
Each vertex represents a substring of length *k* (*k*-mer) in a read. Edges connect vertexes if these are consecutive vertexes, i.e., the last *k*-1 nucleotides in k-mer *u* are the same as the as the first *k*-1 nucleotides of k-mer *v.*

Software: Velvet, SOAPdenovo, SPAdes

Most widely used approaches.

Objective: represent every possible *k*-mer present in the genome!





imed Research Institute for Medicines

Assembly Methods

Three major methods for assembly:

iii) de Bruijn Graph

Connecting the nodes:

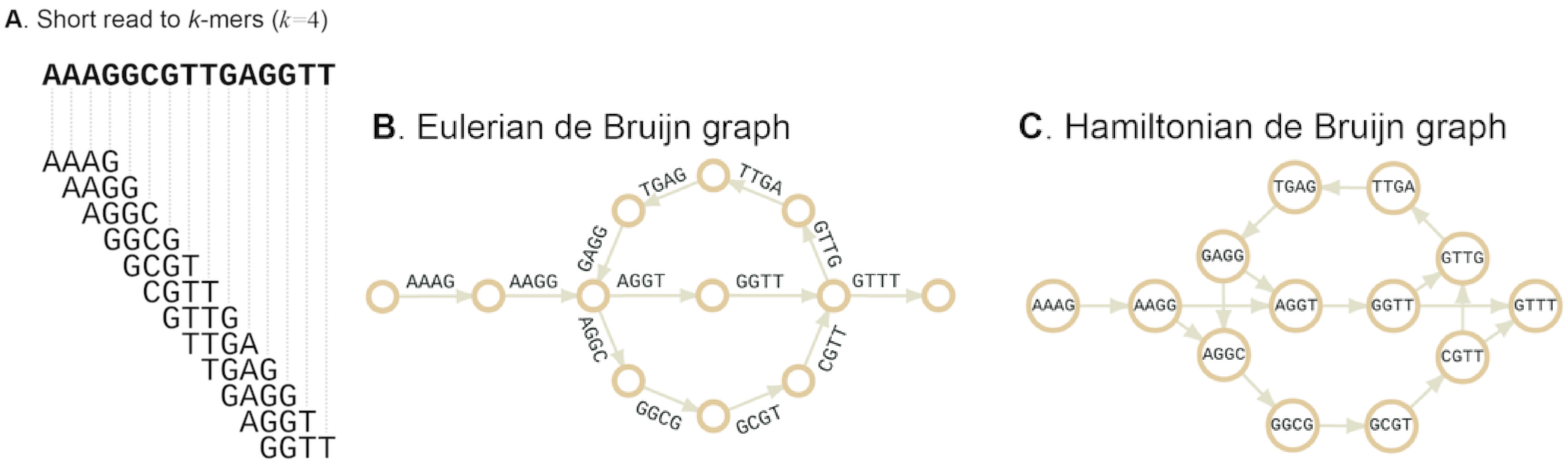
•*Hamiltonian path*: visits **every vertex in the graph** (exactly once, because it is a path)

•*Eulerian trail*: visits **every edge in the graph** exactly once (because it is a trail, vertices may well be crossed more than once.)

William Hamilton

Leonhard Euler

Nicolaas de Bruijn



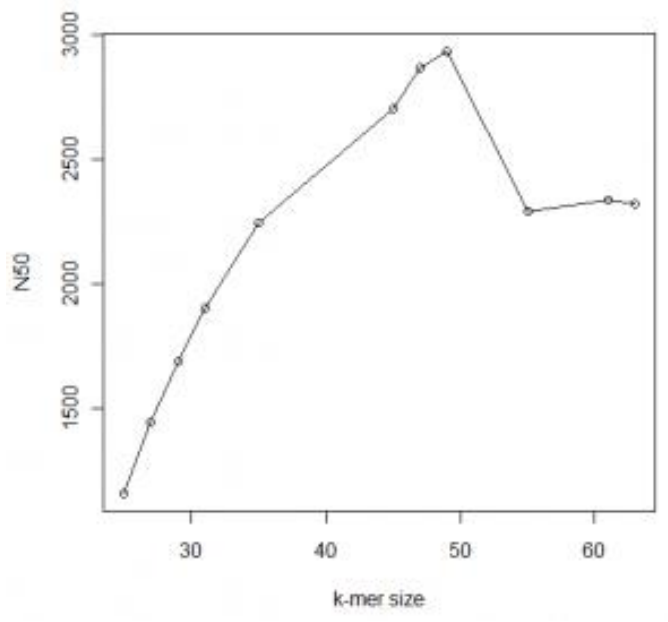
Assembly Methods

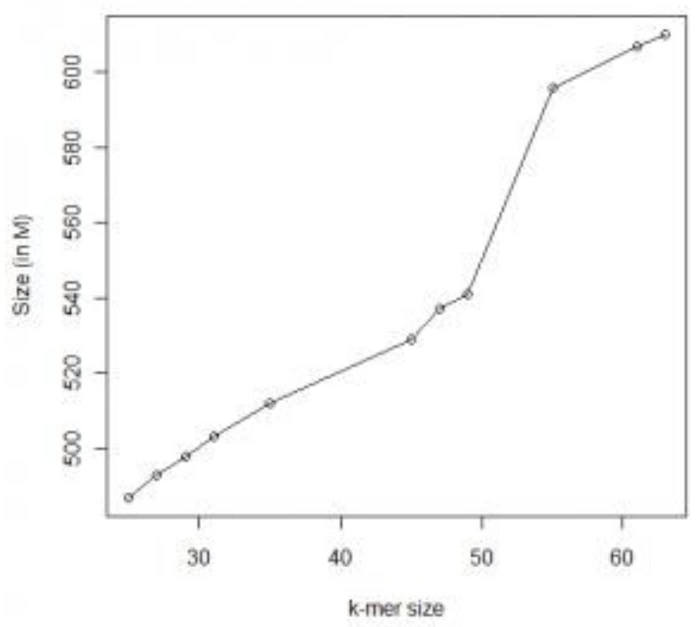
Three major methods for assembly:

iii) de Bruijn Graph

Choice of the k-mer length: always below the read-length of your data!

The k-mer length should be an odd number: avoid palindromes – an odd-sized k-mer cannot form palindromes when reverse-complemented!





https://homolog.us/blogs/genome/2012/10/10/multi-kmer-de-bruijn-graphs/

De Bruijn Graphs - Exercise

Let’s consider the following reads:

ATGCTA
GCTAGC
TAGCAC
CTAGCA

TGC GCT CTA

ATG

TAG

CAC

AGC

GCA

1 List all 3bp k-mers
2 Establish links (edges) between k-mers differing by k-1 nucleotides
3 Visit all nodes and use the minimal path length

De Bruijn Graphs - Exercise

Let’s consider the following reads:

ATGCTA

GCTAGC

TAGCAC

CTAGCA

$$\begin{tikzcd}
ATG \arrow[r] & TGC \arrow[r] \arrow[dd] & GCT \arrow[r] & CTA \arrow[d] \\
 & & & TAG \arrow[dl] \\
CAC & GCA \arrow[l] & AGC \arrow[l] \arrow[uu] &
\end{tikzcd}$$

1 List all 3bp k-mers

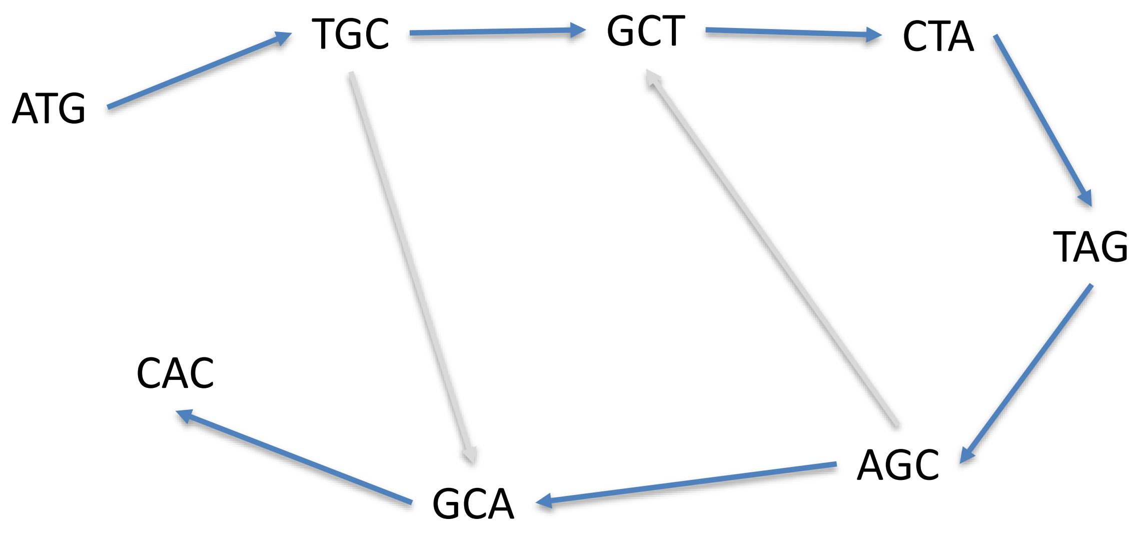
2 Establish links (edges) between k-mers differing by k-1 nucleotides

3 Visit all nodes only once

De Bruijn Graphs - Exercise

Let's consider the following reads:

ATGCTA
GCTAGC
TAGCAC
CTAGCA



ATGCTAGCAC

1 List all 3bp k-mers
2 Establish links (edges) between k-mers differing by k-1 nucleotides
3 Visit all nodes only once

De Bruijn Graphs - Exercise

Let's consider the following reads:

ATGCTA

GCTAGC

TAGCAC

CTAGCA

The Eulerian Path

$$\begin{tikzcd}
AT \arrow[r, "ATG"] & TG \arrow[r, "TGC"] & GC \arrow[r, "GCT"] & CT \arrow[r, "CTA"] & TA \arrow[r, "TAG"] & AG \arrow[r, "AGC"] & GC \arrow[r, "GCA"] & CA \arrow[r, "CAC"] & AC
\end{tikzcd}$$

1 List all 3bp k-mers

2 Define nodes between edges

3 Visit all edges only once

De Bruijn Graphs - Exercise

Let’s consider the following reads:

ATGCTA

GCTAGC

TAGCAC

CTAGCA

The Eulerian Path

$$\begin{tikzcd}
AT \arrow[r, "ATG"] & TG \arrow[r, "TGC"] & GC \arrow[r, "GCT"] & CT \arrow[r, "CTA"] & TA \arrow[r, "TAG"] & AG \arrow[r, "AGC"] & GC \arrow[r, "GCA"] & CA \arrow[r, "CAC"] & AC
\end{tikzcd}$$

1 List all 3bp k-mers

2 Define nodes between edges

3 Visit all edges only once

De Bruijn Graphs - Exercise

Let's consider the following reads:

ATGCTA

GCTAGC

TAGCAC

CTAGCA

The Eulerian Path

$$\begin{tikzcd}
AT \arrow[r, "ATG"] & TG \arrow[r, "TGC"] & GC \arrow[r, "GCT"] & CT \arrow[r, "CTA"] & TA \arrow[r, "TAG"] & AG \arrow[r, "AGC"] & GC \arrow[r, "GCA"] & CA \arrow[r, "CAC"] & AC
\end{tikzcd}$$

$$\begin{tikzcd}
AT \arrow[r, "ATG"] & TG \arrow[r, "TGC"] & GC \arrow[r, "GCT"] \arrow[dr, "GCA"'] & CT \arrow[r, "CTA"] & TA \arrow[r, "TAG"] & AG \arrow[lll, bend right, "AGC"'] \\
 & & & CA \arrow[r, "CAC"] & AC &
\end{tikzcd}$$

ATGCTAGCAC

1 List all 3bp k-mers

2 Define nodes between edges

3 Visit all edges only once

Evaluating and Comparing Assemblies

Metrics to Evaluate and Compare Assemblies:

Evaluating an assembly can be reference-free or comparing to a reference genome**!**

Purpose:

1. Assess the individual quality of an assembly;
2. Compare assemblers.

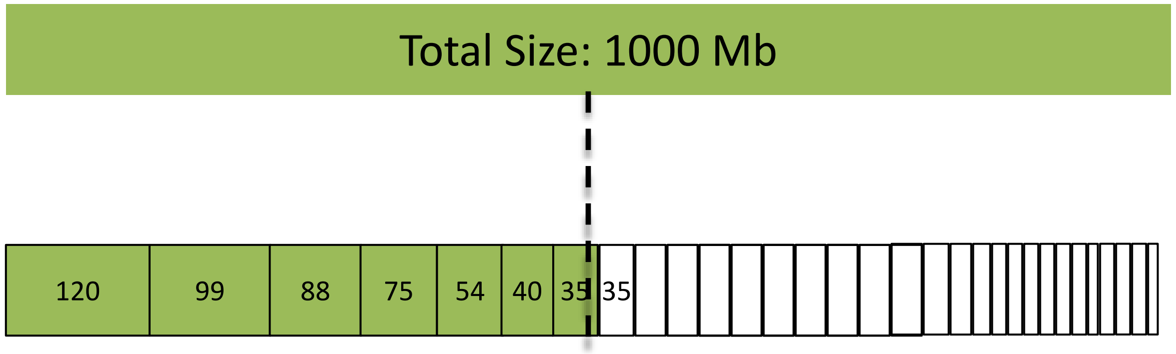
Metrics commonly used:
- Number of contigs/scaffolds
- Total length of the assembly
- Length of the largest contig/scaffold
- Percentage of gaps in scaffolds ('N')
- N50/NG50 of contigs/scaffolds
- Number of predicted genes
- Number of core single-copy genes

Software:
QUAST – allows comparison against reference

BUSCO – evaluates the presence of core single-copy orthologous genes

CheckM - estimates of genome completeness and contamination by using collocated sets of genes that are ubiquitous and single-copy within a phylogenetic lineage

N50 and L50 and other metrics



N50 – shortest contig length spanning the midpoint of the assembly length (after sorting from largest to smallest contig);

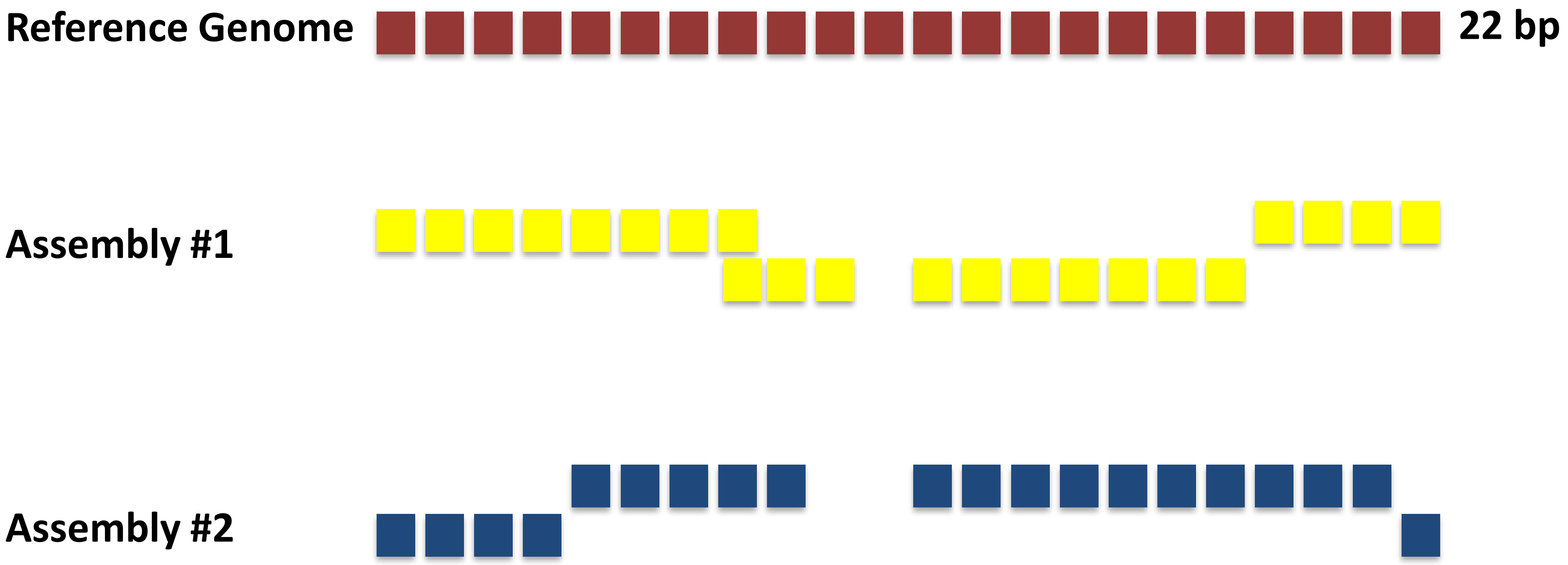
NG50 – shortest contig length spanning the midpoint of the estimated genome size (after sorting from largest to smallest contig);

L50 – number of contigs necessary to span the midpoint of the assembly length (after sorting from largest to smallest contig)

LG50 – number of contigd necessary to span the midpoint of the estimated genome size (after sorting from largest to smallest contig)

N50 and L50 in this exemple?

Evaluating and Comparing Assemblies

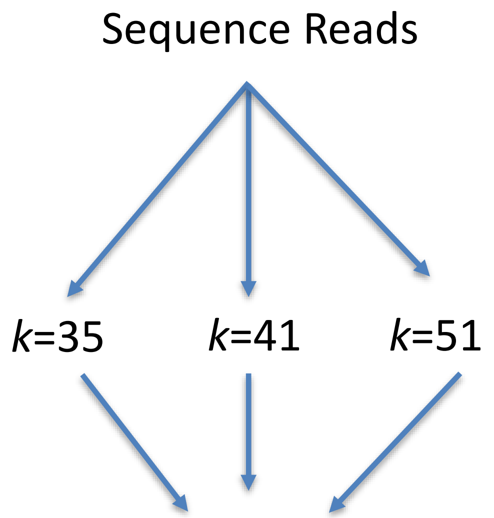


Calculate:	Assembly #1	Assembly #2
N50	**7**	**10**
NG50	7	**5**
Coverage	**95.4%**	**90.9%**
Total Length of Assembly	**22**	**20**

Multi *k-mer* assembly – the way forward

Multi k-mer asssembly:

Multi-k-mer stategies always provide better results than single k-mer



Merge Assemblies -> Velvet+cd-hit+minimus2
Iterative Assembly removing assembled reads – IDBA
Multi-kmer de Bruijn graph - **SPAdes**

SPAdes 3.13.0

imed Research Institute for Medicines

Assembly pipelines

Unicycler

As input, Unicycler takes one of the following:
•Illumina reads from a bacterial isolate (ideally paired-end, but unpaired works too)
•A set of long reads (either PacBio or Nanopore) from a bacterial isolate (uncorrected long reads are fine, though corrected long reads should work too)
•Illumina reads and long reads from the same isolate (best case)

Reasons to use Unicycler:
•It circularises replicons without the need for a separate tool like Circlator.
•It handles plasmid-rich genomes.
•It can use long reads of any depth and quality in hybrid assembly. 10x or more may be required to complete a genome, but Unicycler can make nearly-complete genomes with far fewer long reads.
•It produces an assembly *graph* in addition to a contigs FASTA file, viewable in Bandage.
•It has very low misassembly rates.
•It can cope with very repetitive genomes, such as *Shigella*.
•It's easy to use: runs with just one command and usually doesn't require tinkering with parameters.

Reasons to not use Unicycler:
•You're assembling a eukaryotic genome or a metagenome (Unicycler is designed exclusively for bacterial isolates).
•Your Illumina reads and long reads are from different isolates (Unicycler struggles with sample heterogeneity).
•You're impatient (Unicycler is thorough but not especially fast).

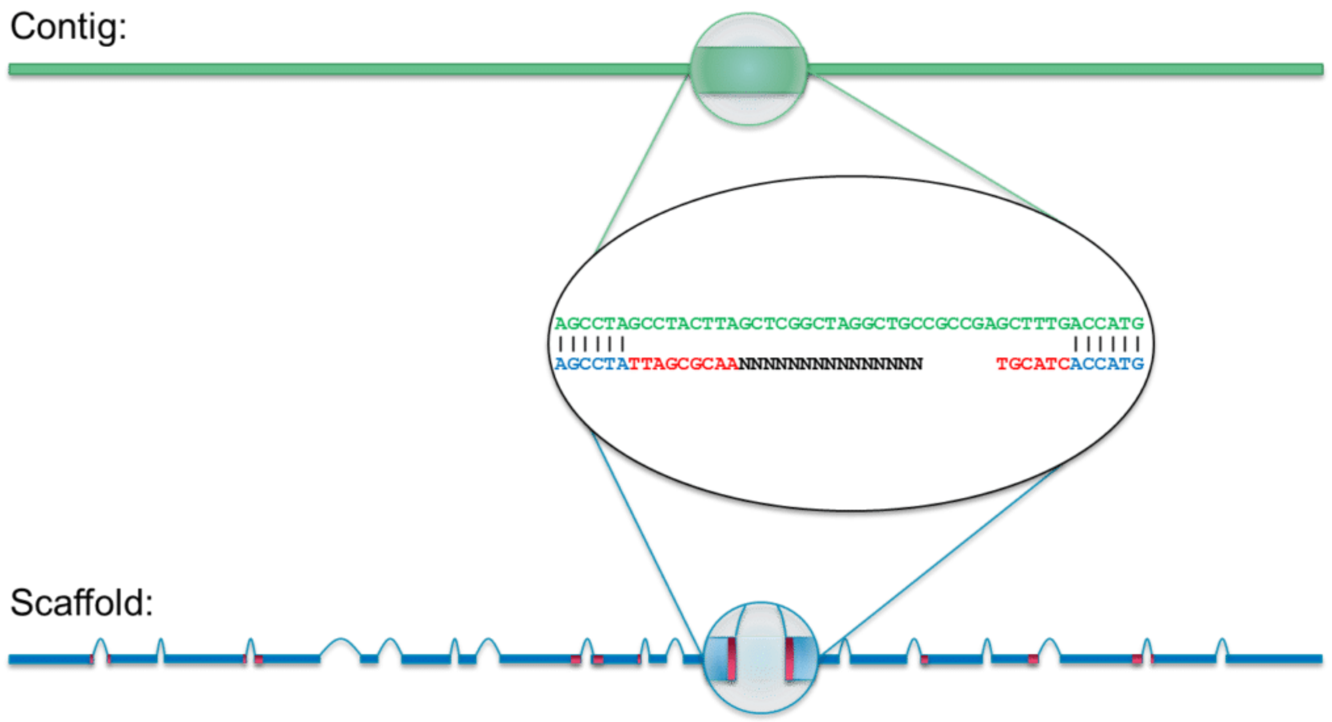
Unicycler does:
- ***Short-read assembly***
- ***Long-read assembly***
- ***Hybrid assembly***

https://github.com/rrwick/Unicycler

Scaffolding

Contigs are continuous stretches of sequence containing only A, C, G, or T bases without gaps.

Scaffolds are created by chaining contigs together ***using additional information*** *about the relative position and orientation of the contigs in the genome*


Contig:
AGCCTAGCCTACTTAGCTCGGCTAGGCTGCCGCCGAGCTTTGACCATG
AGCCTATTAGCGCAANNNNNNNNNNNNNNNN
TGCATCACCATG
Scaffold:

ABACAS

MAUVE

ARCS

imed Research Institute for Medicines

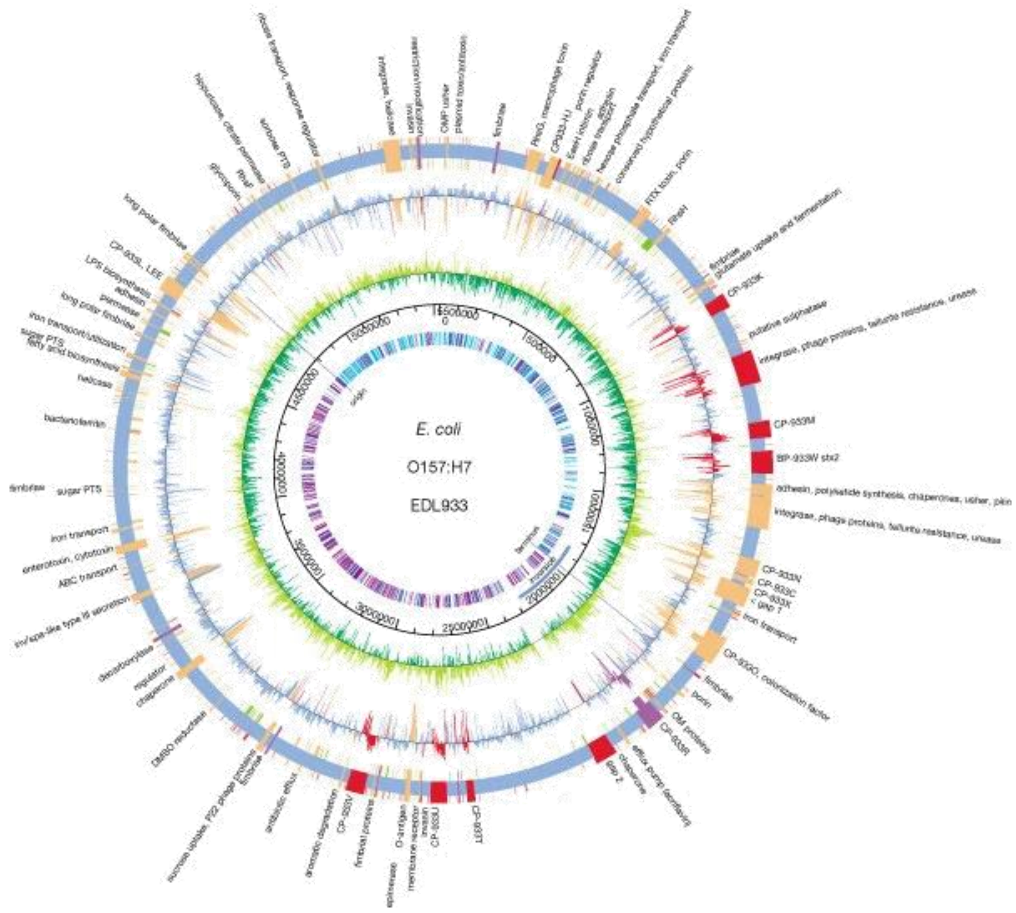
Genome Annotation

Genome annotation refers to the identification of the location of genetic features…

… most often protein-coding genes!

This entails the exact pinpointing of feature coordinates and orientation in the assembled/finished genome.

Easier for bacterial genomes (high coding density ~90%, no introns) – less complex!



Two main approaches: intrinsic (ab initio) vs extrinsic (evidence based)

imed Research Institute for Medicines

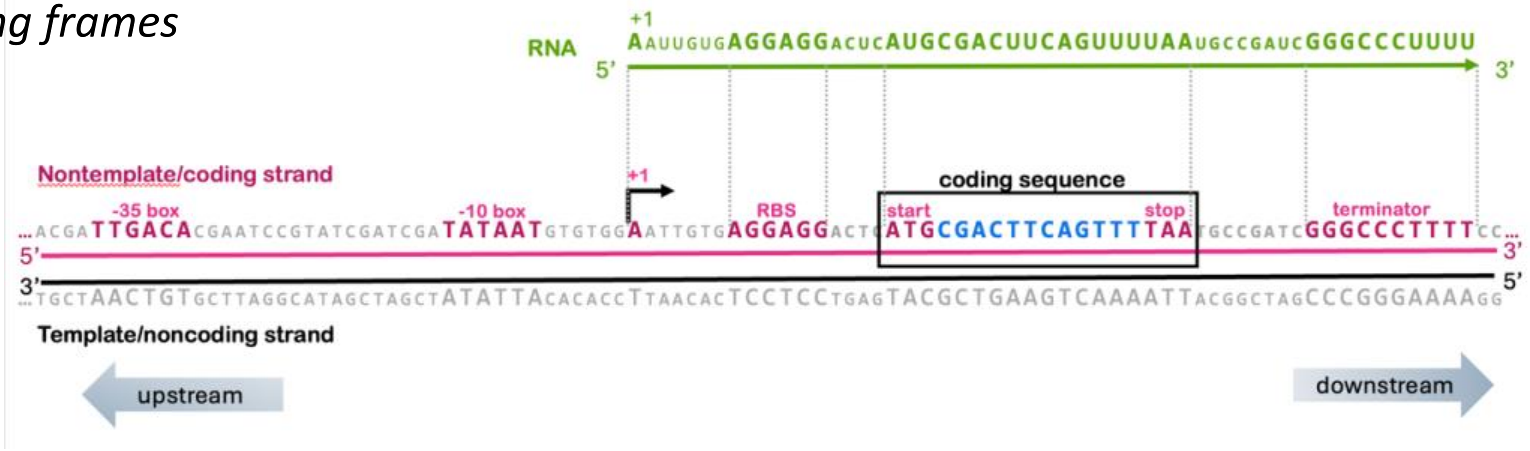
Genome Annotation

Intrinsic (ab initio) gene prediction

- Solely based on DNA sequency without comparison to other sequences or databases

- Search for sequence signals associated with open reading frames

Software/Tools:
GeneMark
GLIMMER
EasyGene
GENSCAN


RNA
+1
AAUUGUGAGGAGGACUCAUGCGACUUCAGUUUUAAUGCCGAUCGGGCCCUUUU
5'
3'
Nontemplate/coding strand
+1
coding sequence
-35 box
-10 box
RBS
start
stop
terminator
...ACGATTGACACGAATCCGTATCGATCGATATAATGTGTGGAATTGTGAGGAGGACTCATGCGACTTCAGTTTTAATGCCGATCGGGCCCTTTTCC...
5'
3'
3'
5'
...TGCTAACTGTGCTTAGGCATAGCTAGCTATATTACACACCTTAACACTCCTCCTGAGTACGCTGAAGTCAAAATTACGGCTAGCCCGGGAAAAGG
Template/noncoding strand
upstream
downstream

Extrinsic – Evidence based gene prediction

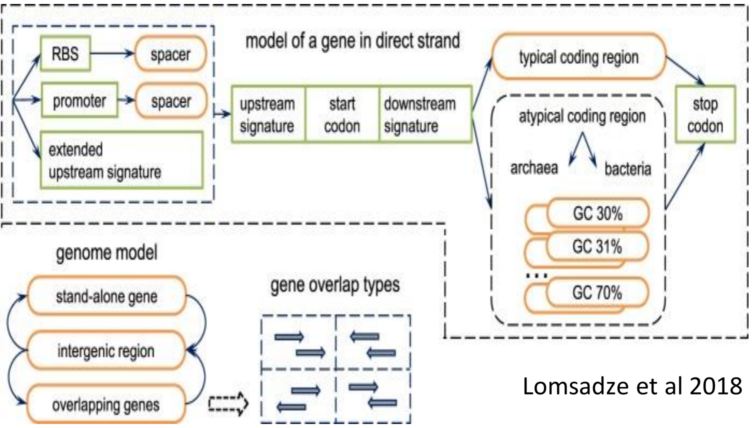
- Homology-based and comparative approach

- Comparison with databases of curated protein families or pre-existing genus/species-specific databases

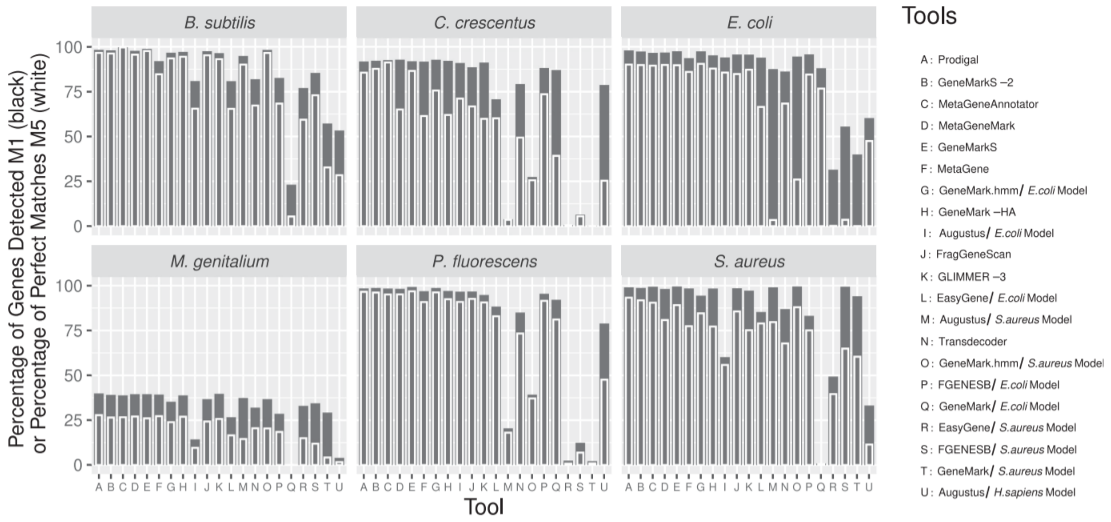
- Can be supported via BLASTX!

Genome Annotation

There are multiple tools for gene prediction, often implementing complex statistical models (Hidden Markov Models) and Machine-Learning algorithms – its performance varies across organism



Lomsadze et al 2018



Dimonaco et al 2022 Bioinformatics

Some of the most widely known tools:

•**Prodigal**: Very fast (E. coli K12 ~10s), unsupervised machine learning – no training, handles draft assemblies and metagenomes.

•**GeneMarkS-2**: More advanced version of GeneMark, specifically for prokaryotes, this new version has the capability for self-training, used at NCBI, incorporates the detection of atypical coding regions;

•**Glimmer3**: Interpolated Harkov Models, useful for Bacteria, Archaea and viruses, developed and used at TIGR.

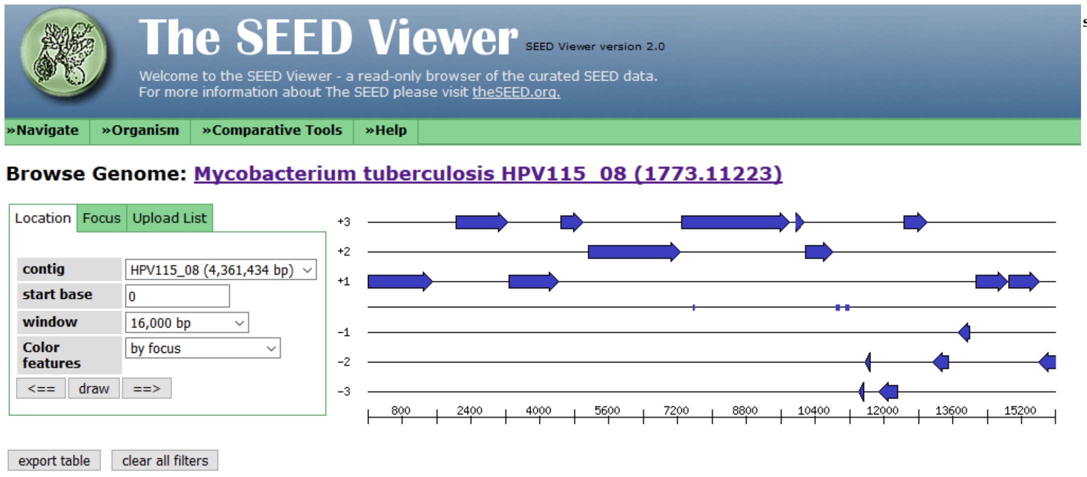
Genome Annotation - RAST

RAST

Web server for prokaryotic genome annotation – user friendly!

Initially calls tRNA and rRNA genes and subsequently calls protein-coding genes via implementation of GLIMMER2;

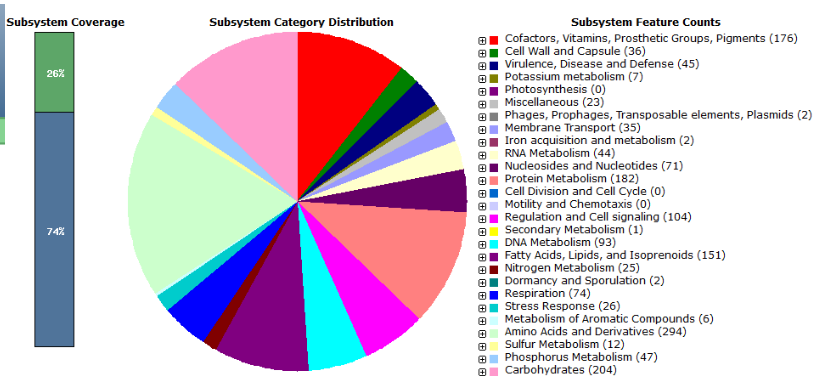
Able to establish phylogenetic context – via comparison of universal FIGfams and subsequent targeteted search across closely related genomes.



Welcome to the SEED Viewer - a read-only browser of the curated SEED data.
For more information about The SEED please visit theSEED.org.

Browse Genome: Mycobacterium tuberculosis HPV115_08 (1773.11223)

contig	HPV115_08 (4,361,434 bp)
start base	0
window	16,000 bp
Color features	by focus

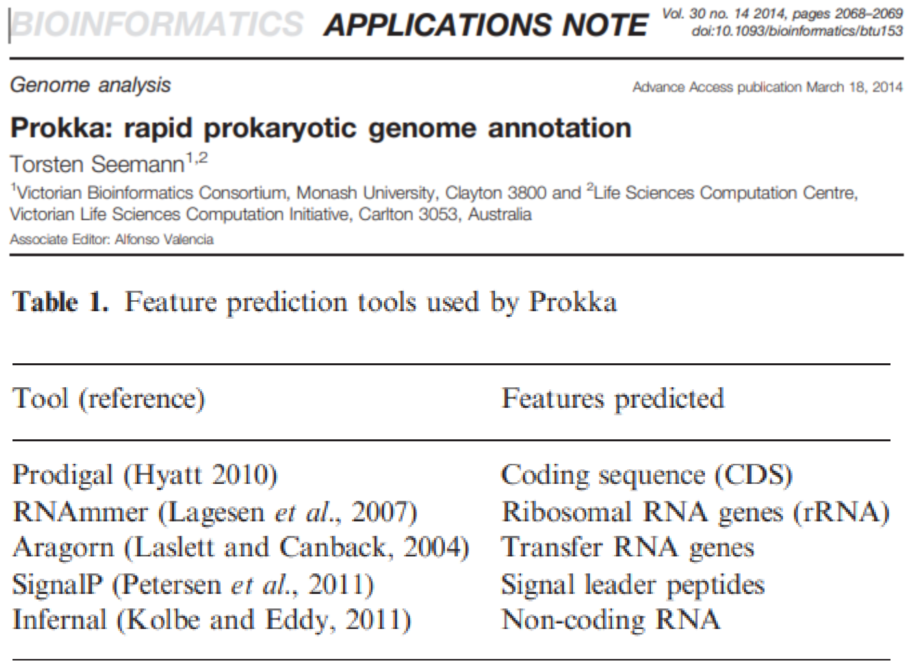


imed Research Institute for Medicines

Genome Annotation - Prokka

Prokka

"Prokka coordinates a suite of existing software tools to achieve a rich and reliable annotation of genomic bacterial sequences. Where possible, it will exploit multiple processing cores, and a typical bacterial genome can be annotated in ~10 min on a quad core desktop computer. It is well suited to iterative models of sequence analysis and integration into genomic software pipelines."



BIOINFORMATICS APPLICATIONS NOTE Vol. 30 no. 14 2014, pages 2068–2069
doi:10.1093/bioinformatics/btu153

Genome analysis Advance Access publication March 18, 2014

Prokka: rapid prokaryotic genome annotation

Torsten Seemann[1,2]
[1]Victorian Bioinformatics Consortium, Monash University, Clayton 3800 and [2]Life Sciences Computation Centre, Victorian Life Sciences Computation Initiative, Carlton 3053, Australia
Associate Editor: Alfonso Valencia

Table 1. Feature prediction tools used by Prokka

Tool (reference)	Features predicted
Prodigal (Hyatt 2010)	Coding sequence (CDS)
RNAmmer (Lagesen *et al.*, 2007)	Ribosomal RNA genes (rRNA)
Aragorn (Laslett and Canback, 2004)	Transfer RNA genes
SignalP (Petersen *et al.*, 2011)	Signal leader peptides
Infernal (Kolbe and Eddy, 2011)	Non-coding RNA

(1) An optional user-provided set of annotated proteins, searched using BLAST+ blastp (Camacho et al., 2009).

(2) All bacterial proteins in UniProt (Apweiler et al., 2004) that have real protein or transcript evidence and are not a fragment. This is 16 000 proteins, and typically covers 450% of the core genes in most genomes. BLAST+ is used for the search.

(3) All proteins from finished bacterial genomes in RefSeq <u>for a specified genus (optional)</u>. This captures domain-specific naming, and the databases vary in size and quality, depending on the popularity of the genus.

(4) A series of hidden Markov model profile databases, including Pfam (Punta et al., 2012) and TIGRFAMs (Haft et al., 2013). This is performed using hmmscan from the HMMER 3.1 package (Eddy, 2011).

(5) If no matches can be found, label as 'hypothetical protein'.

Genome Annotation - Prokka

Prokka

Performance and output files

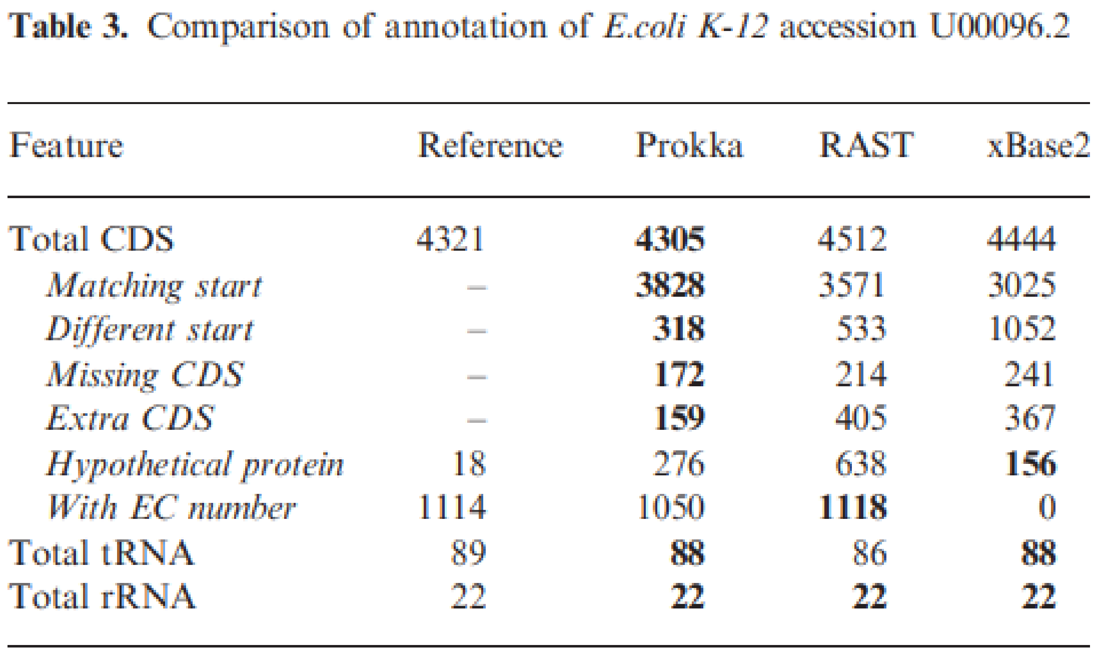


Table 3. Comparison of annotation of *E.coli K-12* accession U00096.2

Feature	Reference	Prokka	RAST	xBase2
Total CDS	4321	**4305**	4512	4444
Matching start	–	**3828**	3571	3025
Different start	–	**318**	533	1052
Missing CDS	–	**172**	214	241
Extra CDS	–	**159**	405	367
Hypothetical protein	18	276	638	**156**
With EC number	1114	1050	**1118**	0
Total tRNA	89	**88**	86	**88**
Total rRNA	22	**22**	**22**	**22**

Prokka produces a full set of output files that enable submission to the NCBI

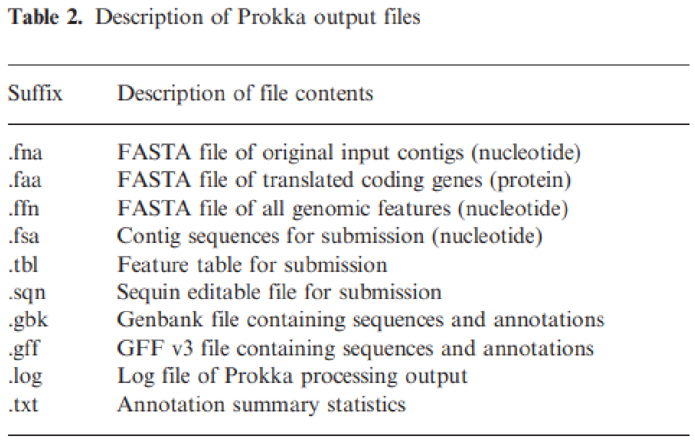


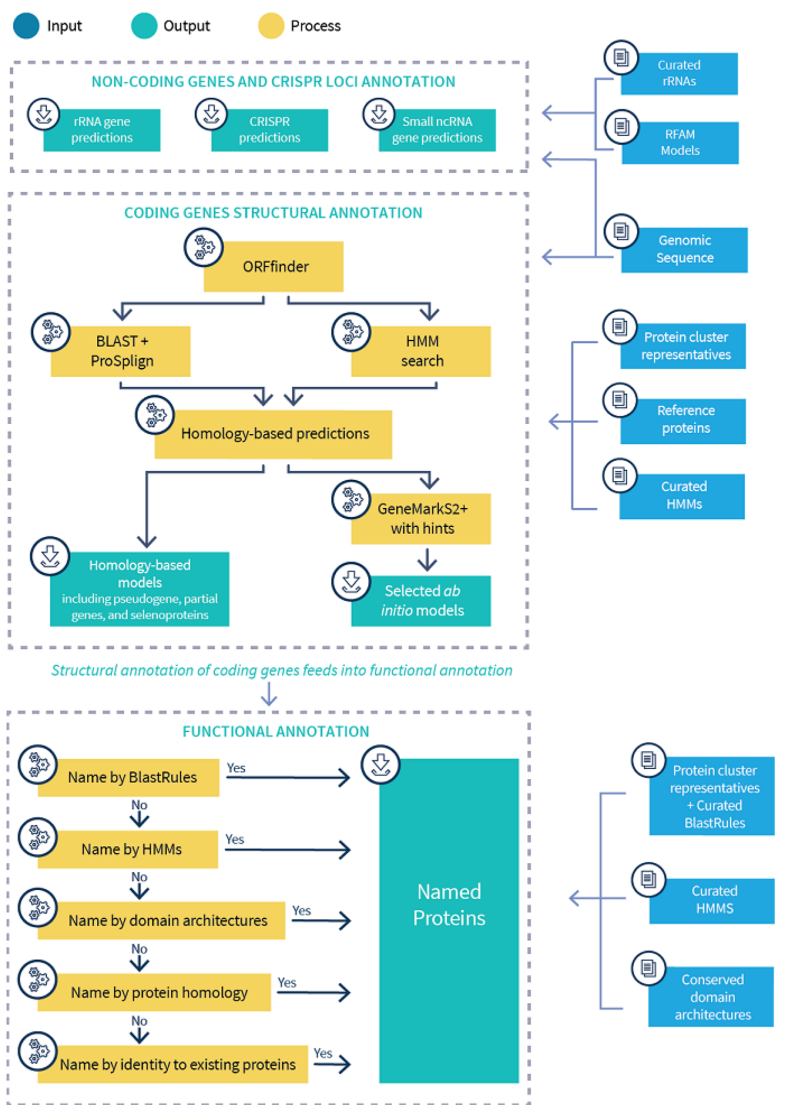
Table 2. Description of Prokka output files

Suffix	Description of file contents
.fna	FASTA file of original input contigs (nucleotide)
.faa	FASTA file of translated coding genes (protein)
.ffn	FASTA file of all genomic features (nucleotide)
.fsa	Contig sequences for submission (nucleotide)
.tbl	Feature table for submission
.sqn	Sequin editable file for submission
.gbk	Genbank file containing sequences and annotations
.gff	GFF v3 file containing sequences and annotations
.log	Log file of Prokka processing output
.txt	Annotation summary statistics

Performance for an E. coli genome (well studied) shows an overall better performance in comparison to RAST or xBase2.

Genome Annotation - PGAP

NCBI – Prokaryotic Genome Annotation Pipeline



Input
Output
Process
NON-CODING GENES AND CRISPR LOCI ANNOTATION
rRNA gene predictions
CRISPR predictions
Small ncRNA gene predictions
Curated rRNAs
RFAM Models
Genomic Sequence
CODING GENES STRUCTURAL ANNOTATION
ORFfinder
BLAST + ProSplign
HMM search
Homology-based predictions
GeneMarkS2+ with hints
Homology-based models including pseudogene, partial genes, and selenoproteins
Selected ab initio models
Protein cluster representatives
Reference proteins
Curated HMMs
Structural annotation of coding genes feeds into functional annotation
FUNCTIONAL ANNOTATION
Name by BlastRules
Name by HMMs
Name by domain architectures
Name by protein homology
Name by identity to existing proteins
Yes
No
Named Proteins
Protein cluster representatives + Curated BlastRules
Curated HMMS
Conserved domain architectures

Combines *ab initio* gene prediction algorithmns with homology based algorithms

Multi-level approach that enables detection of:

- Protein-coding genes
- structural RNAs
- tRNAs
- small RNAs
- Pseudogenes
- control regions
- direct and inverted repeats
- insertion sequences
- transposons and other mobile elements

Now available as a command-line utility (Docker Container) locally deployable!