

Question: is *de novo* genome
assembly a solved problem
with long reads, yet?

Rayan Chikhi

Institut Pasteur & CNRS

CGSI 2019

Answer: No

Answer: No

Thank you! Any questions?

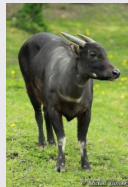
Acknowledgements: Pierre Marijon, Jean-Stéphane Varré, Antoine Limasset, Camille Marchet, Sergey Nurk, Marco Previtali, Paul Medvedev, Shaun Jackman, Guillaume Rizk, Adam Phillippy

Hello

- New group leader at Institut Pasteur, Paris
- *Algorithms and data structures for biological sequences*

Contributions:

- Methods around k -mers
 - ▶ Minia, KmerGenie, DSK, BCALM2
- Genome assemblies



@RayanChikhi on Twitter
<http://rayan.chikhi.name>

Genome assembly

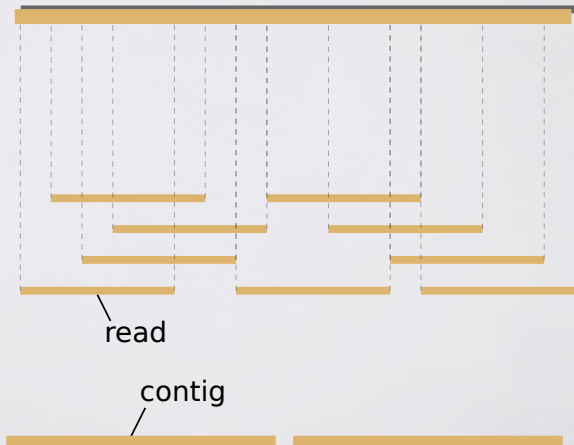
genome
(unknown)

sequenced reads:

overlapping sub-sequences,
covering the genome
redundantly



assembly
hypothesis of
the genome



The many applications of assembly

- ▶ **Reconstruct** genome
- ▶ transcriptome
- ▶ metagenome
- ▶ genes from different taxa
- ▶ Find novel **insertions**
- ▶ **SNPs** in non-model organisms
- ▶ **cell-free DNA** SVs
- ▶ Pangenomics
- ▶ ...



Happy b-day whole-genome assembly



(Staden 1979) *“With modern fast sequencing techniques and suitable computer programs it is now possible to sequence whole genomes without the need of restriction maps.”*

(Adapted from A. Phillippy’s talk, RECOMB-Seq’19)

A short algorithmic history of genome assembly



A short algorithmic history of genome assembly



- Shortest Common Superstring
- Greedy algorithms

A short algorithmic history of genome assembly



- Shortest Common Superstring
- Greedy algorithms
- String graphs and de Bruijn graphs, both introduced at DIMACS in 1994

A History of DNA Sequence Assembly, G. Myers, 2016

Modern genome assembly: graphs

1. Construct a graph
2. Nodes are reads (or k -mers)
3. Edges are overlaps

Theorists will say..

[Nagarajan 09]

4. Return a path of *minimal length* that traverses **each node at least once**.



Genome assembly is **NP-hard**.

[Medvedev, Brudno 2007]

If all **repeats** are **longer** than reads,
Genome assembly is **polynomial**. (!)

[Nagarajan, Pop 2009]

If all repeats are either **shorter** than reads,
or are **spanned** by reads,
Genome assembly is **polynomial**, *and* with a **unique** solution.

[Nagarajan, Pop 2009]

[Bresler, Bresler, Tse 2013]




But, in practice

- **Illumina** data: none of the previous formalisms applied
- Because graph often disconnected
- Contigs = unambiguous paths in graph

- **Long reads**: bridge between theory and practice appears possible
- HINGE assembler [Kamath *et al*, 2017]

Recommended reading

Modeling biological problems in computer science: a case study in genome assembly

Paul Medvedev 

Briefings in Bioinformatics, bby003, <https://doi.org/10.1093/bib/bby003>

Published: 30 January 2018 **Article history** ▼

Vertebrate/human genome assembly



Clip slide

40 Years of Genome Assembly: Are We Done Yet?

Adam M. Phillippy
Head, Genome Informatics Section

@aphillippy 

  National Human Genome
Research Institute

The **Forefront**
of **Genomics**

Outlook:

- Vertebrates: some have high repeat %
- Diploidity: still badly handled
- Humans: Telomere2telomere within 2 years

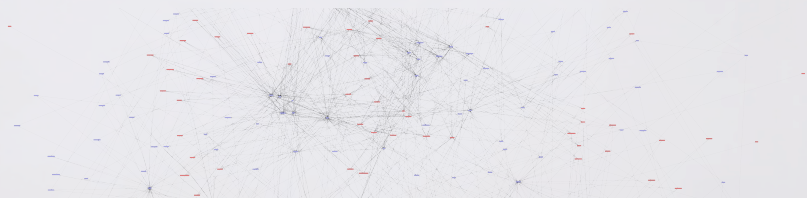
Today: supposedly easy cases



- Small (e.g. bacterial) genomes
- Haploid
- Can we *at least* assemble *that* well now?

Genome assembly software is complex

- Coding:
 - ▶ PhD
 - ▶ or a team of engineers (1-2 years)
- Several not-always-independent components



- Heuristics everywhere

*A good genome assembler is like a good sausage,
you would rather not know what is inside*

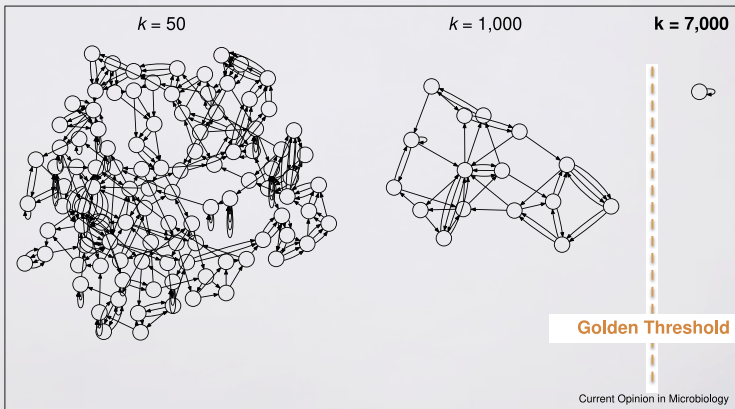
(apocryphal) S. Gnerre, ALLPATHS assembler

Long-read genome assemblers

- HGAP
- Canu
- Falcon
- miniasm
- Unicycler
- MARVEL
- Tulip
- ABruijn
- Hinge
- Flye
- Ra
- Wutabaga2
- Shasta
- Peregrine
- ...

One chromosome = one contig?

Assembly graph of the *E. coli* genome [Koren, Phillippy 2015]:



Slides adapted from P. Marijon, RECOMB-Seq'19

NCTC 3000 database

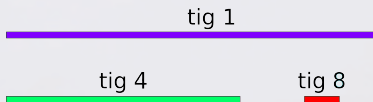
Species	Strain	Sample	Runs	Automated Assembly	Manual Assembly	Manual Assembly Chromosome Contig Number	Manual Assembly Plasmid Contig Number	Manual Assembly Unidentified Contig Number
<i>Achromobacter xylosoxidans</i>	NCTC10807	ERS451415	ERR550491 ERR550506 ERR550507	Pending	EMBL	1	0	0
<i>Budvicia aquatica</i>	NCTC12282	ERS462988	ERR581162	Pending	EMBL	2	0	0
<i>Campylobacter jejuni</i>	NCTC11351	ERS445056	ERR550473 ERR550476	Pending	EMBL	1	0	0
<i>Cedecea neteri</i>	NCTC12120	ERS462978	ERR581152 ERR581168 ERR597265	Pending	EMBL	7	1	0
<i>Citrobacter amalonaticus</i>	NCTC10805	ERS485850	ERR601566 ERR601575	Pending	EMBL	1	2	0
<i>Citrobacter freundii</i>	NCTC9750	ERS485849	ERR601559 ERR601565	Pending	EMBL	1	0	0
<i>Citrobacter koseri</i>	NCTC10849	ERS473430	ERR581173	Pending	EMBL	1	1	0
<i>Corynebacterium diphtheriae</i>	NCTC11397	ERS451417	ERR550510	Pending	EMBL	1	0	0
<i>Cronobacter sakazakii</i>	NCTC11467	ERS462977	ERR581151 ERR581167	Pending	EMBL	4	3	0

599 / 1136 (34 %) assemblies are not single-contig (Feb 2019)

Example (simulated)

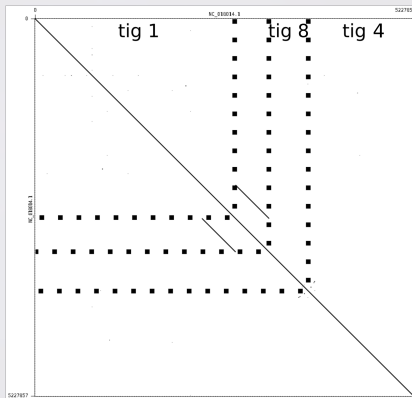
- **Dataset:** *T. roseus* (bacteria), simulated PacBio 20x
- **Assembly:** Canu 1.7

Resulting assembly graph:



Can we recover missing edges between contigs?

Not even a repetition problem..



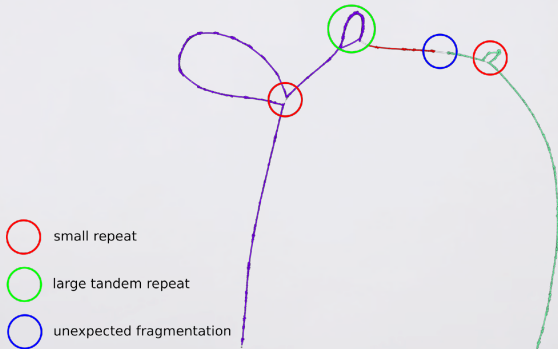
Dotplot of T. roseus genome against itself.

Genome has a 460 kbp tandem repeat.
It explains only 1 of the 2 contigs breaks.

Example (simulated)

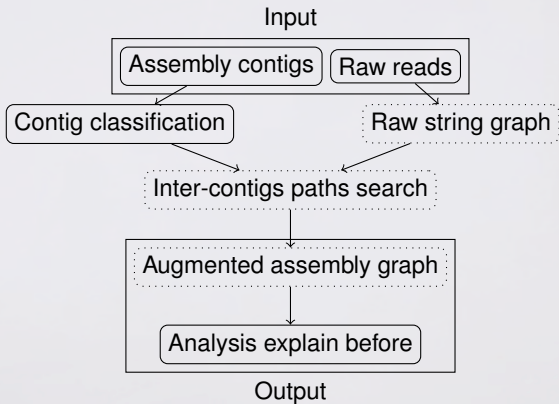
Let's have a look at the original overlap graph:

- nodes → reads
- edges → overlaps



Overlap graph (constructed by `Minimap2`), reads colored by `Canu` contig.

KNOT: Pipeline

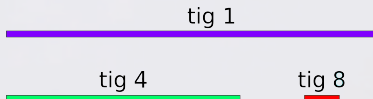


The Augmented Assembly Graph

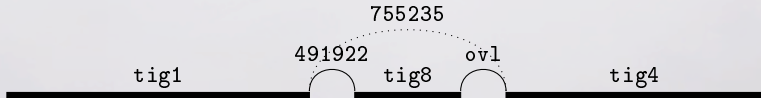
undirected, weighted graph:

- nodes: contigs extremities
- edges:
 - ▶ between extremities of a contig (weight = 0)
 - ▶ paths found between contigs (weight = path length in bp)

From



To

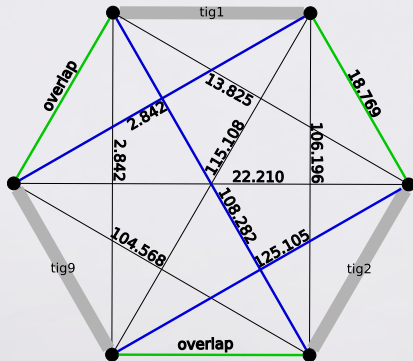


KNOT finds hidden connections between contigs

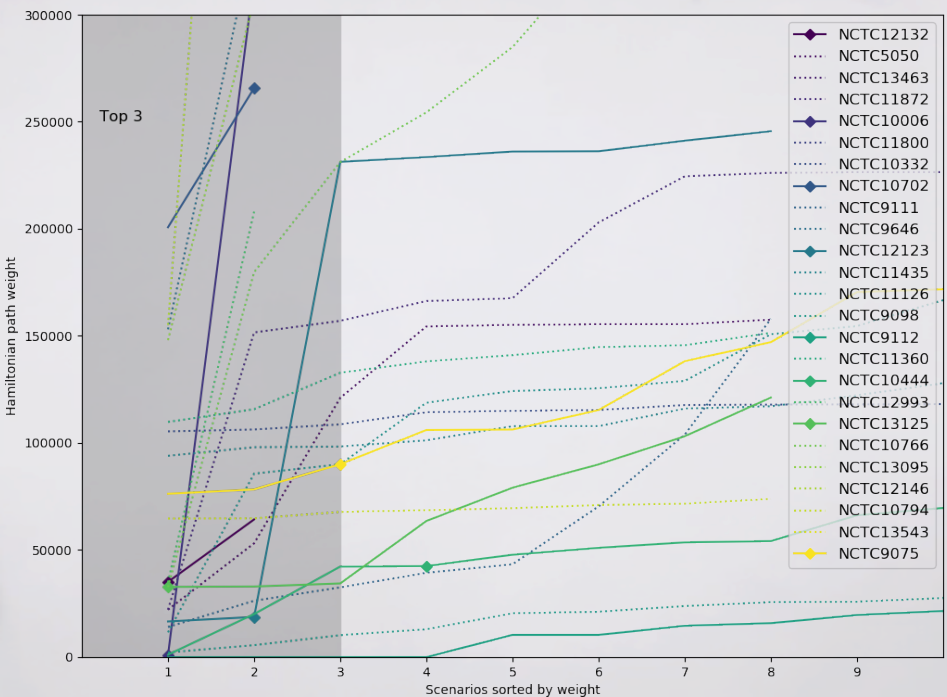
Across 38 datasets:

Mean number of	
Canu contigs	4.32
Dead-ends in Canu contig graph	4.94
Dead-ends in AAG	2.70

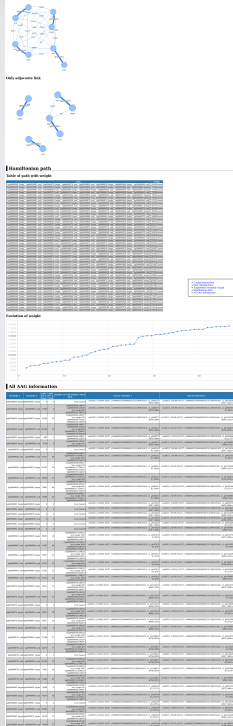
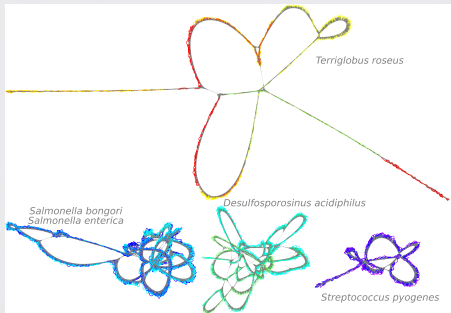
- AAG's are generally complete
- Hamilton walks can be **enumerated**
- Walk weight: sum of edges weights
- **lowest-weight walk** assumed to be the true genome




- Green walk weight: 18,769 bases
- Blue walk weight: 136,229 bases



- Bacterial genome assembly **isn't fully solved**
- **Augmented Assembly Graphs** can help



<https://gitlab.inria.fr/pmarijon/knot>

 @pierre_marion

- Other analysis tool not based on graphs:
- <https://github.com/johnomics/tapestry>

How robust are genome assemblers?



<https://github.com/rrwick/Badread>

Simulates reads with

- chimeras
- low-quality regions
- systematic basecalling errors
- ...

Software testing

Unit test for a single function

```
def AddTest:  
    assert(add(1,1) == 2)
```

Functional test for a whole feature

```
def MapTest:  
    r = mapRead("ACTGATG", genome)  
    assert(r.position = 100000)  
    assert(r.mapping_length = 150)  
    ...
```

Assembly "functional testing"

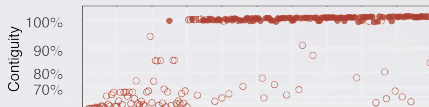
`github.com/rrwick/Long-read-assembler-comparison/`

- Bacterial genome (*C. kerstersii*)
 - ▶ 4 tandem copies of rRNA operon, 20kbp repeat
- Simulated PacBio reads, various
 - ▶ Read lengths
 - ▶ Error rates
 - ▶ Coverages
 - ▶ ...

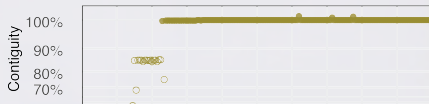
Badread

R. Wick, K. Holt

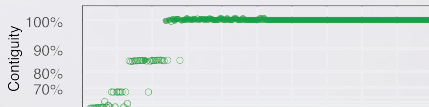
Canu
v1.8



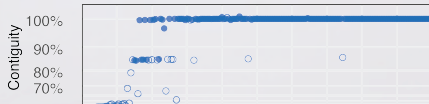
Flye
v2.4.2



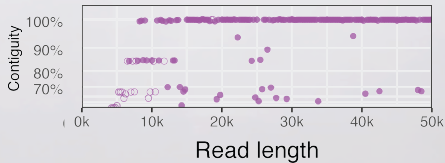
Ra
07364a1



Unicycler
v0.4.7



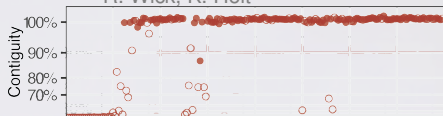
Wtdbg2
v2.4



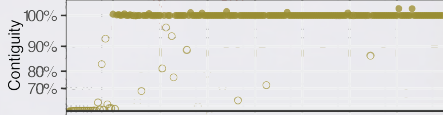
Badread

R. Wick, K. Holt

Canu
v1.8



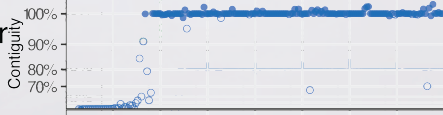
Flye
v2.4.2



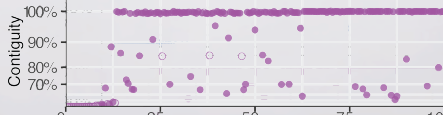
Ra
07364a1



Unicycler
v0.4.7



Wtdbg2
v2.4

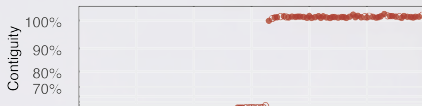


0x 25x 50x 75x 100x
Read depth

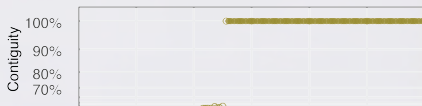
Badread

R. Wick, K. Holt

Canu
v1.8



Flye
v2.4.2



Ra
07364a1



Unicycler
v0.4.7



Wtdbg2
v2.4

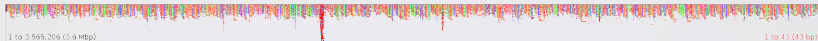


70%

Read identity

Reproducing that benchmark

- Same dataset as the previous benchmark
- Simulated reads using another program (PaSS, 2019)
- 50x depth
- Should assemble fine into 1 contig (..maybe?)

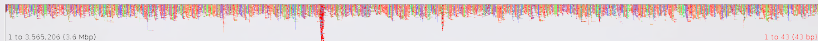


- Same assemblers

	Canu	Flye	Ra	Unicycler	Wtdbg2
# contigs					

Reproducing that benchmark

- Same dataset as the previous benchmark
- Simulated reads using another program (PaSS, 2019)
- 50x depth
- Should assemble fine into 1 contig (..maybe?)



- Same assemblers

	Canu	Flye	Ra	Unicycler	Wtdbg2
# contigs	1 😊	1 😊	2 😐	3 😐	2 😐

Is it a coverage problem?

Is it a coverage problem?

- from 50x coverage



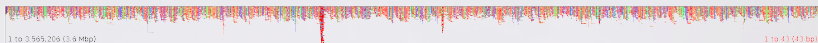
- to 100x coverage



	Canu	Flye	Ra	Unicycler	Wtdbg2
# contigs					

Is it a coverage problem?

- from 50x coverage



- to 100x coverage



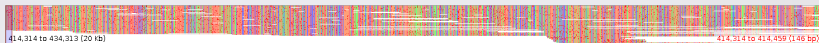
	Canu	Flye	Ra	Unicycler	Wtdbg2
# contigs	1 😊	1 😊	2 😐	3 😐	3 😐

- So, no

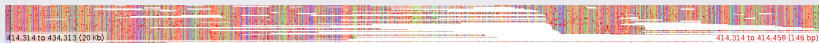
Robustness to coverage drops

- Taking the 50x coverage dataset again
- Simulating a coverage drop to **10x** somewhere

Before:



After:

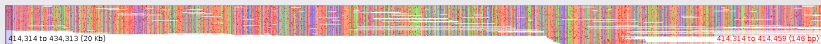


	Canu	Flye	Ra	Unicycler	Wtdbg2
# contigs					

Robustness to coverage drops

- Taking the 50x coverage dataset again
- Simulating a coverage drop to **10x** somewhere

Before:



After:



	Canu	Flye	Ra	Unicycler	Wtdbg2
# contigs	1 😊	1 😊	2 😐	3 😐	2 😐

- Assemblers: all unphased by that drop

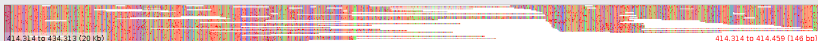
Robustness to heavy coverage drop

- Dropping down to **5x** 😈

10x:



5x:



	Canu	Flye	Ra	Unicycler	Wtdbg2
# contigs					

Robustness to heavy coverage drop

- Dropping down to **5x** 😈

10x:



5x:



	Canu	Flye	Ra	Unicycler	Wtdbg2
# contigs	2 😐	1 😊	3 😐	3 😐	2 😐

- Good job Flye!

Conclusion



- Assembly status: **unsolved**
- Benchmarks: **needed**
- Tools presented here: KNOT, Badread

Thank you! Any questions? (for real now)

Acknowledgements: Pierre Marijon, Jean-Stéphane Varré, Antoine Limasset, Camille Marchet, Brian Bushnell, Sergey Nurk, Marco Previtali, Paul Medvedev, Shaun Jackman, Guillaume Rizk, Ryan Wick, Tablet software, Adam Phillippy