de novo paired-end short reads assembly

Rayan Chikhi
ENS Cachan Brittany
Symbiose, Irisa, France
THESIS FOCUS

- Graph theory for assembly models
- Indexing large sequencing datasets
- Practical implementation in the Monument assembler
Outline

Assemblers

Measures and benchmarks

Monument

Mapsembler

Conclusion
A typical assembly pipeline

1. Raw reads
2. Reads error-correction
3. Single reads indexing
4. Single reads assembly
5. Paired reads scaffolding
6. Scaffolds gap-closing
7. Contigs, scaffolds

- Error-corrector
- Assembler
- Scaffolder
- Gapcloser
REMINDER : ASSEMBLY MODELS

A Read Layout

R₁: GACCTACA
R₂: ACCTACAA
R₃: CCTACAG
R₄: CTACAAGT
A: TACAAGTT
B: ACAAGTTA
C: CAAGTTAG
X: TACAAAGTC
Y: ACAAGTCC
Z: CAAGTCCG

B Overlap Graph

C de Bruijn Graph

0. Figure : Schatz et al. 2010
THE ASSEMBLY PROBLEM

Let $G$ be the string graph\(^1\) of the reads

Assembly Problem: find a generalized Hamiltonian path (visit each node at least once) of minimum length

---

1. overlap graph without transitive edges and contained reads
2. [Myers 2005]
**Why Greedy/OLC/DBG instead of AP?**

- \( R \) number of reads
- \( r \) read length
- \( S \) genome length
- \( k \) overlap length or \( k \)-mer length

**Structure sizes:**

- **Overlap graph**: \(|V| = R, |E| \approx R\), label length \( r \)
- **de Bruijn graph**: \(|V| \approx |E| \approx S\), label length \( k \)
- **Greedy**: array structure, \( \approx S \) elements, \( k \)-mer keys

Practically: \( R >> S \) and \( r > k \).

Overlap graph implementation (Newbler): 4 bytes per read base\(^3\).

Compressed de Bruijn graph: 12 bytes per graph edge\(^4\).

---

\(^3\) J. Knight (Roche). Assembly and Finishing of Large/Complex Genomes. 2009 talk

\(^4\) [Link](http://arxiv.org/pdf/1008.2555v1)
LANDSCAPE OF ASSEMBLY

- Before the Illumina Hi-Seq:
  - **Long reads** (>200bp), **low coverage**
    - string graphs
    - Newbler, Cabog
  - **Short reads** (<100bp), **high coverage**
    - de Bruijn graphs
    - Velvet, SOAPdenovo, AbySS, ALLPATHS

- Now and future: **100 bp reads**, **high coverage**, **mate pairs**
  - which data structure for assemblers?

- New trend: greedy assemblers with ad-hoc structure
  - (Ray, PE-Assembler, Meraculous, Monument)
## Short-Read Assemblers

<table>
<thead>
<tr>
<th>Assembler</th>
<th>Method</th>
<th>Error Corr.</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Euler</td>
<td>de Bruijn</td>
<td>pre-assembly</td>
<td>Pioneer</td>
</tr>
<tr>
<td>Velvet</td>
<td>de Bruijn</td>
<td>in-assembly</td>
<td>(still) Popular</td>
</tr>
<tr>
<td>ABySS, CLC-bio⁵, SOAPdenovo</td>
<td>de Bruijn</td>
<td>in-assembly</td>
<td>Parallel, large genomes</td>
</tr>
<tr>
<td>Allpaths LG</td>
<td>de Bruijn</td>
<td>pre-assembly</td>
<td>Needs short/long inserts</td>
</tr>
<tr>
<td>IDBA⁶</td>
<td>de Bruijn</td>
<td>pre-assembly</td>
<td>Multiple-k</td>
</tr>
<tr>
<td>Cabog, Newbler</td>
<td>String</td>
<td>in-assembly</td>
<td>Long reads</td>
</tr>
<tr>
<td>Ray</td>
<td>de Bruijn</td>
<td>in-assembly</td>
<td>Parallel short/long reads</td>
</tr>
<tr>
<td>PE-Assembler⁷</td>
<td>ad-hoc</td>
<td>pre-assembly</td>
<td>Shorty-like, no graph</td>
</tr>
<tr>
<td>SGA⁸</td>
<td>String</td>
<td>pre-assembly</td>
<td>FM-index, promising</td>
</tr>
</tbody>
</table>

---

8. [https://github.com/jts/sga](https://github.com/jts/sga)
PERSONAL EXPERIENCE (FOR ILLUMINA ASSEMBLY)

General purpose  SOAPdenovo

Best quality  Allpaths-LG

Genome too large  ABySS

Easy to run (once installed)  Ray
DE NOVO METAGENOMIC/RNA ASSEMBLERS

*de novo* metagenomic assemblers:

**Genovo**: Assembles up to $10^5$ 454 reads. Uses a probabilistic model + ICM method.

**MetaVelvet**: based on Velvet

*de novo* RNA assemblers:

**Oases**: Actually a post-processing step for Velvet.

**Trinity**: new name for Ananas

Outline

Assemblers

Measures and benchmarks

Monument

Mapsembler

Conclusion
N50

N50 = Scaffold/contig length at which you have covered 50% of total assembly length

NG50 = Scaffold/contig length at which you have covered 50% of total genome length

Genome

Assembly
OTHER MEASURES

Reference-based:
- **Global accuracy** (% of 10 kbp blocks which align with > 90% identity)\(^{12}\)
- **Coverage** \(^{13}\)
- **Errors**: small substitutions, small indels, chimeric joins \(^{14}\)

Without reference:
- **Internal consistency**: read coherence and happy pairs \(^{15}\)

---

12. Allpaths/bin/AssemblyAccuracy
13. Allpaths/bin/AssemblyCoverage
14. Custom MUMmer-based script
15. AMOS
**DIPLOID ACCURACY MEASURE**

**Scaffolds path N50**

*Adjacency graph* (diploid genome)

[Paten et al., 2011, submitted]

Assembly (red line):

**Scaffold path**: maximal paths with consistent edges

---

Assembly challenges

**Goal:** given a dataset of reads, produce the best possible assembly

---

**Assemblathon**
- Feb 2011
- 100 Mbp diploid genome
- Hi-Seq reads, 80x + mate-pairs
- Organized by UCSC/UCSD
- 17 participants

**dnGASP**
- Mar 2011
- 2 Gbp diploid genome
- Hi-Seq reads, 44x + mate-pairs
- Organized by CNAG
- 9 participants
Assemblathon: Best Scaffold NG50

<table>
<thead>
<tr>
<th>Assembler</th>
<th>NG50 scaffold length</th>
<th>NG50 contig length</th>
</tr>
</thead>
<tbody>
<tr>
<td>G:JGI:1</td>
<td>15913</td>
<td>9073174</td>
</tr>
<tr>
<td>Q:Broad:ALLPATH-LG:1</td>
<td>219906</td>
<td>6396795</td>
</tr>
<tr>
<td>I:CSHL:Celera:1</td>
<td>136900</td>
<td>3254796</td>
</tr>
<tr>
<td>C:EBI:SGA:2</td>
<td>2722</td>
<td>3032585</td>
</tr>
<tr>
<td>E:ABYSS:1</td>
<td>17259</td>
<td>2712723</td>
</tr>
<tr>
<td>K:SOAPdenovo+Seqclean:1</td>
<td>214502</td>
<td>1801023</td>
</tr>
<tr>
<td>H:Monument:3</td>
<td>6849</td>
<td>1421767</td>
</tr>
<tr>
<td>M:OligoZip:1</td>
<td>84611</td>
<td>1417207</td>
</tr>
<tr>
<td>V:Velvet:4</td>
<td>8192</td>
<td>879312</td>
</tr>
<tr>
<td>B:Sanger:Phusion2:1</td>
<td>69204</td>
<td>502551</td>
</tr>
<tr>
<td>J:Iowa:PCAP:1</td>
<td>301691</td>
<td>301691</td>
</tr>
<tr>
<td>A:PE-Assembler:1</td>
<td>29571</td>
<td>57002</td>
</tr>
<tr>
<td>L:Price:1</td>
<td>22716</td>
<td>22716</td>
</tr>
<tr>
<td>W:CLC:1</td>
<td>1281</td>
<td>12187</td>
</tr>
<tr>
<td>N:Cortex:3</td>
<td>9358</td>
<td>9358</td>
</tr>
<tr>
<td>O:Kiki:1</td>
<td>6000</td>
<td>6000</td>
</tr>
<tr>
<td>ID</td>
<td># Contigs</td>
<td>N50</td>
</tr>
<tr>
<td>----</td>
<td>-----------</td>
<td>------------</td>
</tr>
<tr>
<td>Q1</td>
<td>1,946</td>
<td>208,256</td>
</tr>
<tr>
<td>P1</td>
<td>4,566</td>
<td>343,889</td>
</tr>
<tr>
<td>J1</td>
<td>4,791</td>
<td>301,691</td>
</tr>
<tr>
<td>I2</td>
<td>37,571</td>
<td>130,666</td>
</tr>
<tr>
<td>H1</td>
<td>1,798</td>
<td>151,121</td>
</tr>
<tr>
<td>E5</td>
<td>28,683</td>
<td>56,660</td>
</tr>
<tr>
<td>M1</td>
<td>4,477</td>
<td>65,510</td>
</tr>
<tr>
<td>M3</td>
<td>45,200</td>
<td>58,916</td>
</tr>
<tr>
<td>F3</td>
<td>29,300</td>
<td>48,200</td>
</tr>
<tr>
<td>F4</td>
<td>32,134</td>
<td>47,737</td>
</tr>
<tr>
<td>M4</td>
<td>2,672</td>
<td>67,017</td>
</tr>
<tr>
<td>B1</td>
<td>4,503</td>
<td>66,597</td>
</tr>
<tr>
<td>F2</td>
<td>33,437</td>
<td>35,510</td>
</tr>
<tr>
<td>F1</td>
<td>45,487</td>
<td>34,247</td>
</tr>
<tr>
<td>M5</td>
<td>13,998</td>
<td>36,938</td>
</tr>
<tr>
<td>M2</td>
<td>5,780</td>
<td>36,443</td>
</tr>
<tr>
<td>K2</td>
<td>2,796</td>
<td>214,562</td>
</tr>
<tr>
<td>K3</td>
<td>926</td>
<td>216,303</td>
</tr>
<tr>
<td>K1</td>
<td>15,689</td>
<td>209,662</td>
</tr>
<tr>
<td>A1</td>
<td>9,741</td>
<td>25,383</td>
</tr>
</tbody>
</table>

NA50: « alignment » N50

SPA50: scaffold path NA50
# ASSEMBLATHON: RESOURCES

<table>
<thead>
<tr>
<th>Resource</th>
<th>Cpu time (h)</th>
<th>Wall clock (h)</th>
<th>Memory per node (Gb)</th>
</tr>
</thead>
<tbody>
<tr>
<td>H: Monument</td>
<td>2</td>
<td>48</td>
<td>6.3</td>
</tr>
<tr>
<td>G: JGI: Meraculous</td>
<td>6</td>
<td>256</td>
<td>4</td>
</tr>
<tr>
<td>B: Phusion2</td>
<td>6.5</td>
<td>31</td>
<td>100</td>
</tr>
<tr>
<td>A: PE-Assembler</td>
<td>3</td>
<td>384</td>
<td>100</td>
</tr>
<tr>
<td>Q: ALLPATH-LG</td>
<td>12</td>
<td>70</td>
<td>100</td>
</tr>
<tr>
<td>P: SOAPdenovo</td>
<td>24</td>
<td>64</td>
<td>70</td>
</tr>
<tr>
<td>N: Cortex</td>
<td>24</td>
<td>120</td>
<td>64</td>
</tr>
<tr>
<td>I: Celera</td>
<td>24</td>
<td>120</td>
<td>100</td>
</tr>
<tr>
<td>O: Kiki</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L: Price</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>J: PCAP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>K: SOAPdenovo+Seqclean</td>
<td>368</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td>F: ABySS+Anchor</td>
<td>345</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td>M: OligoZip</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C: EBI: SGA</td>
<td>300</td>
<td>128</td>
<td></td>
</tr>
<tr>
<td>E: ABySS</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>D: Sanger: SGA</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
dngasp : preliminary stats

Edited, probably not public
**What’s next? Assemblathon 2**

_Maylandia zebra_, Illumina short and long inserts, ≈ 100 bp reads, 192x coverage, 1 Gbp genome.

_Red tailed boa constrictor_, Illumina short and long inserts, ≈ 100 bp reads

_Common pet parakeet_, Illumina short and long inserts, 454

Submissions deadline: Sept. 1st.
OUTLINE

Assemblers

Measures and benchmarks

Monument

Mapsembler

Conclusion
**Motivation for a New Model**

Common problems with current assembly methods:
- Usually high memory footprint
- Computation-intensive for longer genomes
- Error correction dilemma
  - pre-assembly: no efficient algorithm, loss of information
  - in-assembly: creates too many vertices in graphs

In the future:
- Longer read lengths: de Bruijn-based assemblers will be inadequate
- Higher throughput: overlap graph-based assemblers will be inadequate
**Paired Assembly Problem**

*Paired string graph*: also represents paired links between reads.

*Path-strings*: string spelled by a path. Gaps allowed, e.g. $ab\diamond defgh$.

*Paired Assembly problem*: find a minimum weight generalized H.P. s.t. path-string satisfies pairing constraints

17. to appear in WABI 2011 proceedings
Theoretical motivation: can one build scaffolds directly? 
e.g. localized assembly

Monument algorithm

Starting read
Theoretical motivation: can one build scaffolds directly?  
e.g. localized assembly

Monument algorithm

- Starting read
- Classical overlap-based extension
- Traditional assembly
Theoretical motivation: can one build scaffolds directly? e.g. localized assembly

Monument algorithm

Starting read

Classical overlap-based extension

Paired extension

Traditional assembly
Theoretical motivation: can one build scaffolds directly? e.g. localized assembly

Monument algorithm

Starting read

Classical overlap-based extension

Traditional assembly

Paired extension

Repeat overlap extension, etc.
Monument algorithm

Theoretical motivation: can one build scaffolds directly?
e.g. localized assembly

Starting read

Classical overlap-based extension

Paired extension

Repeat overlap extension, etc..

Gapfilling using paired links

Traditional assembly
## Bacterial Results

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Software</th>
<th>Contig N50 (Kbp)</th>
<th>Scaffold N50 (Kbp)</th>
<th>Longest scaffold (Kbp)</th>
<th>Coverage (%)</th>
<th>Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experimental (1)</td>
<td>Monument</td>
<td>38.0</td>
<td>101.8</td>
<td>236.0</td>
<td>96.4</td>
<td>96.7</td>
</tr>
<tr>
<td></td>
<td>Velvet</td>
<td>26.3</td>
<td>95.3</td>
<td>267.9</td>
<td>96.9</td>
<td>99.1</td>
</tr>
<tr>
<td></td>
<td>Ray</td>
<td>69.5</td>
<td>87.3</td>
<td>174.4</td>
<td>97.4</td>
<td>98.4</td>
</tr>
<tr>
<td>Simulated with variants (2)</td>
<td>Monument</td>
<td>113.3</td>
<td>134.1</td>
<td>340.5</td>
<td>91.0</td>
<td>95.0</td>
</tr>
<tr>
<td></td>
<td>Velvet</td>
<td>30.8</td>
<td>132.6</td>
<td>327.2</td>
<td>87.9</td>
<td>92.3</td>
</tr>
<tr>
<td></td>
<td>Ray</td>
<td>10.2</td>
<td>10.2</td>
<td>41.2</td>
<td>89.2</td>
<td>100.0</td>
</tr>
</tbody>
</table>

Resources for bacterial assembly: 7 minutes, 0.5 GB RAM.
MONUMENT HIGHLIGHTS

- First software able to do targeted assembly of scaffolds
  - Collaboration for targeted assembly of chilean grape SVs
- Human assembly in < 80 GB RAM
- 1 Gbp/day on 6 nodes cluster
OUTLINE

Assemblers

Measures and benchmarks

Monument

Mapsembler

Conclusion
Mapsembler: focus on **specific** information

Biological information is in the reads -
Peterlongo HTS2011
Mapsembler: focus on « known » information

• A fragment = starter.

• A set of NGS reads

Biological information is in the reads -
Peterlongo HTS2011

• Mapsembler, targeted assembly of larges genomes on a desktop computer – INRIA Report, March 2011.
1. Is the starter coherent with a subset of reads?

   ATGCGGCATTGCA
   CATGGACATGCAGGC
   TGGACATGCAGGCAT
   GCATTGCATAGC
   GCATAGCTGACT

2. If yes:
   1. give the context: contig containing the starter
      ...CAACGGACGCATATGGACATGCGGCATTGCA
   2. give the complex context: graph

   Biological information is in the reads - Peterlongo HTS2011
• Validate a $k$-mer assembly
• Is this gene has an homologue is this read set?
• RNA seq: is this splicing event present/absent?
• Metagenomic: is this enzyme present in this sample?
• Remove “annoying” reads (contamination, symbiont, ...)
• Enrich unmappable reads
• ...

• “Zero” memory. (small) desktop computer
Phase 1a: map reads on starter
(each read: at most $d$ substitutions)

Biological information is in the reads -

Peterlongo HTS2011
Phase 1b: generates starters “read-coherent” at most d substitutions.
Phase 2a: Extensions
Extend each Starter’

New extension

Biological information is in the reads -
Peterlongo HTS2011
Phase 2b: Extensions
extend each extensions...

New extension...

Biological information is in the reads -
Peterlongo HTS2011
Phase 2b: Extensions
when do we stop?

• V1: If case of branching: 2 possible extensions

• Outputs fasta file

Biological information is in the reads -
Peterlongo HTS2011
Phase 2b: Extensions
when do we stop?

- V2: Never.: will generate a tree...

Biological information is in the reads -

Peterlongo HTS2011
Phase 2b: Extensions
when do we stop?

• V2: generate a tree...

• ... and finally a graph:

• Outputs a XGMML graph (Cytoscape)
Mapsembler: some examples

Exon skipping (drosophila)

Biological information is in the reads -
Peterlongo HTS2011
SNPs (drosophila)

Biological information is in the reads -

Peterlongo HTS2011
Detection of 24 new candidate fusion genes implicated in human breast cancer.

**Fig. 3.** Extension graph of exon VAPB (20:56,962,507-56,966,573). The dark blue node is the starter. The light blue nodes are exons on chromosome 17, gene IKZF3. Nodes with red borders correspond to fusion exon found in [2]. Green nodes correspond to exons with no fusion genes, also found in normal tissue.
MAPSEMBLER DEMO MAYBE?
Let’s conclude

• Non model species
• Avoid assembly:
  – Keep all information
  – Desktop computer
• Answer specific question
• Complementary to whole genome assembly approaches

Biological information is in the reads -
Peterlongo HTS2011
Infos and downloads: http://alcovna.genouest.org/

Biological information is in the reads - Peterlongo HTS2011
SOFTWARE FROM SYMBIOSE NGS TEAM

- **GASSST**
  Fast, parallel reads alignment with arbitrary gap length
- **KissSNP**
  Localize SNPs between two datasets SNPs a reference
- **Alcovna project**
  Report RNA splicing events without a reference
- **MAPPI project**
  Efficient intersection of two (metagenomic) reads sets
ACKNOWLEDGEMENTS

- Dominique Lavenier, PhD advisor
- Pierre Peterlongo, collaboration on Mapsembler
- Symbiose team
- Biogenouest platform (http://www.biogenouest.org/)
Symbiose

IRISA, INRIA/CNRS/ENS Cachan, Rennes

- NGS algorithms
  - sequence alignment
  - assembly (since 2008)
  - targeted assembly (since 2010)
  - metagenomic analysis (since 2010)

- Workflow and parallelization

- Biologicals networks and models

- Proteins: structures and grammars

- BioGenouest platform
Stratégies d’assemblage

- Graphe des reads chevauchants (overlap graph)