READ MAPPING ON DBG

Antoine Limasset
Pierre Peterlongo

http://arxiv.org/abs/1505.04911
What is a genome?

Biology

fasta

> My full genome
ATCGCTTAAACGGCTCTTTGGCCGGGGGTGCG
TTTGATGCTACGGCGATTCTTGGAGAGCCA
GCAGCGACTGCAAATGTGAGATCAGAGTAA
TATTAGCAAGCGATAAGTGAGATTCAACCTTC
ATACAGATCTAGAGTCTAAAGCAGTGATCT
CCCGCGTGCGAGATAAAAATACTAGGTAAC
TAGAGGGACTGCGACGTTCTAAACGTTGGT
CCGTCAGAAGCGCCATCCAGGATCACGTTA
CCCCGAAAAAAAGATATCAGGAGCTCTCCT
CCTCTGCAGTCAGGTCTATAGAAACTACAG
GACTAACCTTCCTGGCAACCGGGAGGTGGG
AATCCGTCACATATGAGAAGGTATTTGCCC
GATMTCATATCTGGCAGGTCTTTT
What is a genome?

- Linear vision limited to:
  - Small &
  - unique (one individual) &
  - simple species &
  - well assembled
  (ie. *Acinetobacter baylyi* ADP1)

- More complex vision (graph) for
  - All other cases (almost all).
A genome = A graph. Demo

A De Bruijn graph shows the structure of the gene, that we are going to stretch manually to see the gene in a linear structure.
A genome = A graph. Why?

- Biological:
  - SNPs
  - Variants
  - Individuals

- Technical:
  - Sequencing errors
  - Repeats longer than data (reads or kmers)
  - Inexact repeats
Sequence mapping

**Basic local alignment search tool**
SF Altschul, W Gish, W Miller, EW Myers... - Journal of molecular..., 1990 - Elsevier
A new approach to rapid sequence comparison, basic local alignment search tool (BLAST), directly approximates alignments that optimize a measure of local similarity, the maximal segment pair (MSP) score. Recent mathematical results on the stochastic properties of ...
Cité 55145 fois Autres articles Les 115 versions Citer Enregistré

**Gapped BLAST and PSI-BLAST: a new generation of protein database search programs**
SF Altschul, TL Madden, AA Schäffer... - Nucleic acids..., 1997 - Oxford Univ Press
Abstract The BLAST programs are widely used tools for searching protein and DNA databases for sequence similarities. For protein comparisons, a variety of definitional algorithmic and statistical refinements described here permits the execution time of the ...
Cité 55046 fois Autres articles Les 144 versions Citer Enregistré
Sequence mapping / comparison

- The basical tool
  - Knowledge inference
  - Population genetics
  - Genome re-sequencing
  - Sequence selection
  - Quantification
  - Taxonomy
  - Phylogeny
  - Assembly
  - Correction
- The three magical words:
  - Agronomy
  - Health
  - Environment
Mapping on a flat sequence?

- Flat sequence good and representative:
  - Okay

- Else, limitations:
  - Wrong mapping = mapped in a wrong position
    - Because of flat sequence consensus
  - Not mapped
    - Because of flat sequence consensus
    - Because of end of contig (assembly context)

Contig 1

Contig 2

Contig 3

Unmappable read
Our proposal: mapping on a [c]dbg

- Fixes the mapping issues
- Provides additonal information on mapped sequences (reads)
  - Compression
  - Correction
- Provides additonal information on reference graph
  - Quantification
  - Assembly
  - Phasing
dbg and cdbg

• **dbg (k=4)**

- AATC → ATCG → TCGA → CGAT → GATT → ATTC → TTCA
- AATC → ATCC → TCGT → CGTT → GTTT → TTTC

• **cdbg**

- AATC
- ATCC
- TCGT
- CGTT
- GTTT
- TTTC

(each node contains a « unitig »)
Cool but hard to map reads on dBG

NP-hard proof

- Travelling Salesman Problem
- Fixed length Travelling Salesman Problem
- Graph Read Mapping Problem
- De Bruijn Graph Read Mapping Problem
We can deal with heuristics

- Our proposal: seed and extend algorithm

1. **Seed:**
   - Seeds are the unitigs overlaps

2. **Extend:**
   - A. extend extremities
   - B. extend between the detected overlaps.
Implementation: B-GREAT

- **Input**
  - Ref: untigs
  - Query: reads

- **Output**
  - For each read: its path in the dbg

- **Usage:**
  1. Count k-mers (keep only solid ones): **DSK**
  2. Create unitigs on solid k-mers: **B-CALM**
  3. Call **B-GREAT**.
### Results

<table>
<thead>
<tr>
<th>Graph</th>
<th>Mapped set (nb reads)</th>
<th>MS</th>
<th>Nb mapped on branching parts of DBG</th>
<th>time (RPS)</th>
<th>memory</th>
<th>Nb reads fully mapped on unitigs</th>
<th>Overall nb mapped reads</th>
</tr>
</thead>
</table>
| Coli        | SRR959239  
(5,372,832) | G  | 687,997                            | 2m2        | 15 MB  | 4,295,627                        | 4,983,624              |
|             |                      | C  | 688,933                            | 1h24       |        |                                  |                        |
| El_cpx      | SRR065390  
(67,617,092) | G  | 44,686,355                         | 3h08       | 1.16GB | 15,592,918                       | 60,279,273             |
|             |                      |    | (66.09%)                           |            |        | (23.06%)                         | (89.15%)               |
| El_norm     |                      |    | 13,994,715                         | 1h55       | 380MB  | 48,442,146                       | 62,436,861             |
|             |                      |    | (20.70%)                           |            |        | (71.64%)                         | (92.34%)               |
| El_sparse   |                      |    | 10,467,181                         | 1h15       | 210MB  | 52,288,269                       | 62,755,450             |
|             |                      |    | (15.48%)                           |            |        | (77.33%)                         | (92.81%)               |
| El_norm     | SRR1522085  
(22,509,110) |    | 3,523,416                          | 12min25s   | 380MB  | 16,682,194                       | 20,205,610             |
|             |                      |    | (15.65%)                           |            |        | (74.11%)                         | (89.77%)               |
| Human       | SRR345593  
and SRR345594 
(2,967,536,821) | G  | 1,004,182,363                      | 11h48      | 18 GB  | 1,533,456,046                    | 2,537,638,409          |
|             |                      |    | (33.84%)                           |            |        | (51.67%)                         | (85.51%)               |
Next usage - phasing

- How to represent results?
- How to use them?
- How to adapt tools?
- How to consider missing values
- How to theoretically compare with variable string graphs
- How to deal with errors
  - Small and rare (illumina)
  - Large and numerous (PacBio)
Next usage – Compress & correct reads

<table>
<thead>
<tr>
<th>Usage</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compression</td>
<td>• Unitigs</td>
</tr>
<tr>
<td></td>
<td>• For each read: mapping path</td>
</tr>
<tr>
<td></td>
<td>• Mapping errors</td>
</tr>
<tr>
<td>Correction</td>
<td>• Mapping errors</td>
</tr>
<tr>
<td>Compression &amp; correction</td>
<td>• Unitigs</td>
</tr>
<tr>
<td></td>
<td>• For each read: mapping path</td>
</tr>
</tbody>
</table>
Conclusion

• Why are we the first?
  • Formalism
  • Model and implementation

• Encouraging results
  • Time similar to bowtie
  • Most of reads map only on branching parts
    • Eg. C. Elegans (23% on unitigs + 66% on branches)

• Limitations
  • Low error rate
  • Only substitutions
  • Everything remains to be done
    • (compression, correction, phasing, indexation, …)