Assembly in the Clouds
Michael Schatz

October 13, 2010
Beyond the Genome
Shredded Book Reconstruction

- Dickens accidentally shreds the first printing of *A Tale of Two Cities*
  - Text printed on 5 long spools

It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, ...

- How can he reconstruct the text?
  - 5 copies × 138,656 words / 5 words per fragment = 138k fragments
  - The short fragments from every copy are mixed together
  - Some fragments are identical
Greedy Reconstruction

The repeated sequence make the correct reconstruction ambiguous

- It was the best of times, it was the [worst/age]

Model sequence reconstruction as a graph problem.
de Bruijn Graph Construction

- $D_k = (V, E)$
  - $V =$ All length-$k$ subfragments ($k < l$)
  - $E =$ Directed edges between consecutive subfragments
    - Nodes overlap by $k-1$ words

- Locally constructed graph reveals the global sequence structure
  - Overlaps between sequences implicitly computed

Original Fragment  Directed Edge

It was the best of  It was the best  was the best of

- de Bruijn, 1946
- Idury and Waterman, 1995
- Pevzner, Tang, Waterman, 2001
It was the best of times, it was the worst of times.

After graph construction, try to simplify the graph as much as possible.
de Bruijn Graph Assembly

It was the best of times, it

of times, it was the

it was the worst of times, it

the age of foolishness

the age of wisdom, it was the

After graph construction, try to simplify the graph as much as possible
Short Read Assembly

- **Genome assembly as finding an Eulerian tour of the de Bruijn graph**
  - Human genome: >3B nodes, >10B edges

- **The new short read assemblers require tremendous computation**
  - Velvet (Zerbino & Birney, 2008) serial: > 2TB of RAM
  - ABBySS (Simpson et al., 2009) MPI: 168 cores x ~96 hours
  - SOAPdenovo (Li et al., 2010) pthreads: 40 cores x 40 hours, >140 GB RAM
Hadoop MapReduce
http://hadoop.apache.org

• MapReduce is the parallel distributed framework invented by Google for large data computations.
  – Data and computations are spread over thousands of computers, processing petabytes of data each day (Dean and Ghemawat, 2004)
  – Indexing the Internet, PageRank, Machine Learning, etc…
  – Hadoop is the leading open source implementation
    • GATK is an alternative implementation specifically for NGS

• Benefits
  – Scalable, Efficient, Reliable
  – Easy to Program
  – Runs on commodity computers

• Challenges
  – Redesigning / Retooling applications
    – Not Condor, Not MPI
    – Everything in MapReduce
Hadoop for NGS Analysis

**Quake**
Quality-aware error correction of short reads
Correct 97.9% of errors with 99.9% accuracy

(Kelley, Schatz, Salzberg, 2010*)
http://www.cbcb.umd.edu/software/quake/

**CloudBurst**
Highly Sensitive Short Read Mapping with MapReduce
100x speedup mapping on 96 cores @ Amazon

(Schatz, 2009)
http://cloudburst-bio.sf.net

**Crossbow**
Searching for SNPs with Cloud Computing
SNPs in 4 hours and ~$85 per human genome

(Langmead, Schatz, Lin, Pop, Salzberg, 2009)
http://bowtie-bio.sf.net/crossbow/

**Myrna**
Cloud-scale differential gene expression for RNA-seq
Expression of 1.1 billion RNA-Seq reads in <2 hours for ~$66

(Langmead, Hansen, Leek, 2010)
http://bowtie-bio.sf.net/myrna/
Graph Compression

- After construction, many edges are unambiguous
  - Merge together compressible nodes
  - Graph physically distributed over hundreds of computers

Design Patterns for Efficient Graph Algorithms in MapReduce.
Warmup Exercise

- Who here was born closest to October 13?
  - You can only compare to 1 other person at a time

Find winner among 64 teams in just 6 rounds
Fast Path Compression

Challenges
- Nodes stored on different computers
- Nodes can only access direct neighbors
- No "Tournament Bracket"

Randomized List Ranking
- Randomly assign $\heartsuit$/$\clubsuit$ to each compressible node
- Compress $\heartsuit\rightarrow\clubsuit$ links

Initial Graph: 42 nodes

Randomized Speed-ups in Parallel Computation.
Fast Path Compression

Challenges
- Nodes stored on different computers
- Nodes can only access direct neighbors
- No "Tournament Bracket"

Randomized List Ranking
- Randomly assign H/T to each compressible node
- Compress H→T links

Randomized Speed-ups in Parallel Computation.
Fast Path Compression

Challenges
- Nodes stored on different computers
- Nodes can only access direct neighbors
- No "Tournament Bracket"

Randomized List Ranking
- Randomly assign $\text{H}/\text{T}$ to each compressible node
- Compress $\text{H}\rightarrow\text{T}$ links

Round 2: 15 nodes (64% savings)

Randomized Speed-ups in Parallel Computation.
Fast Path Compression

Challenges

- Nodes stored on different computers
- Nodes can only access direct neighbors
- No "Tournament Bracket"

Randomized List Ranking

- Randomly assign $\ obsolete$ to each compressible node
- Compress $\ obsolete$ links

Round 2: 8 nodes (81% savings)

Randomized Speed-ups in Parallel Computation.
Fast Path Compression

Challenges

- Nodes stored on different computers
- Nodes can only access direct neighbors
- No "Tournament Bracket"

Randomized List Ranking

- Randomly assign $\overline{H}$/$\overline{T}$ to each compressible node
- Compress $\overline{H} \rightarrow \overline{T}$ links

Round 3: 6 nodes (86% savings)

Randomized Speed-ups in Parallel Computation.
Fast Path Compression

Challenges
- Nodes stored on different computers
- Nodes can only access direct neighbors
- No "Tournament Bracket"

Randomized List Ranking
- Randomly assign $\mathcal{H}$/$\mathcal{T}$ to each compressible node
- Compress $\mathcal{H}\rightarrow\mathcal{T}$ links

Performance
- Compress all chains in $\log(S)$ rounds

Round 4: 5 nodes (88% savings)

Randomized Speed-ups in Parallel Computation.
Node Types

- Isolated nodes (10%)
- Tips (46%)
- Bubbles/Non-branch (9%)
- Dead Ends (.2%)
- Half Branch (25%)
- Full Branch (10%)

(Chaisson, 2009)
De novo bacterial assembly

- **Genome:** *E. coli* K12 MG1655, 4.6Mbp
- **Input:** 20.8M 36bp reads, 200bp insert (~150x coverage)
- **Preprocessor:** Quake Error Correction

### Assembly of Large Genomes with Cloud Computing.
## E. coli Assembly Quality

Incorrect contigs: Align at < 95% identity or < 95% of their length

<table>
<thead>
<tr>
<th>Assembler</th>
<th>Contigs ≥ 100bp</th>
<th>N50 (bp)</th>
<th>Incorrect contigs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contrail PE</td>
<td>300</td>
<td>54,807</td>
<td>4</td>
</tr>
<tr>
<td>Contrail SE</td>
<td>529</td>
<td>20,062</td>
<td>0</td>
</tr>
<tr>
<td>SOAPdenovo PE</td>
<td>182</td>
<td>89,000</td>
<td>5</td>
</tr>
<tr>
<td>ABySS PE</td>
<td>233</td>
<td>45,362</td>
<td>13</td>
</tr>
<tr>
<td>Velvet PE</td>
<td>286</td>
<td>54,459</td>
<td>9</td>
</tr>
<tr>
<td>EULER-SR PE</td>
<td>216</td>
<td>57,497</td>
<td>26</td>
</tr>
<tr>
<td>SSAKE SE</td>
<td>931</td>
<td>11,450</td>
<td>38</td>
</tr>
<tr>
<td>Edena SE</td>
<td>680</td>
<td>16,430</td>
<td>6</td>
</tr>
</tbody>
</table>

It was the best of times, it was the worst of times, it was the age of...
De novo Assembly of the Human Genome

- **Genome**: African male NA18507 (SRA000271, Bentley et al., 2008)
- **Input**: 3.5B 36bp reads, 210bp insert (~40x coverage)

Assembly of Large Genomes with Cloud Computing.
One more thing...
Variations and de Bruijn Graphs

Searching for de novo mutations in the families of 3000 autistic children.

- Assemble together reads from mom, dad, affected & unaffected children
- Look for sequence paths unique to affected child

MRC1LI
Summary

• Surviving the data deluge means computing in parallel
  – Cloud computing is an attractive platform for large scale sequence analysis and computation

• Significant obstacles ahead
  – Time and expertise required for development
  – Transfer time
  – Privacy / security requirements
  – Price
  – What are the alternatives?

• Emerging technologies are a great start, but we need continued research
  – A word of caution: new technologies are new
Acknowledgements

Steven Salzberg

Mihai Pop

Jimmy Lin

Ben Langmead

Dan Sommer

David Kelley
Thank You!

http://schatzlab.cshl.edu

@mike_schatz
**K-mer Counting**

- Application developers focus on 2 (+1 internal) functions
  - **Map**: input \(\rightarrow\) key:value pairs
  - **Shuffle**: Group together pairs with same key
  - **Reduce**: key, value-lists \(\rightarrow\) output

Map, Shuffle & Reduce
All Run in Parallel

---

**ATGAACCTTA**

- (ATG:1) (ACC:1)
- (TGA:1) (CCT:1)
- (GAA:1) (CTT:1)
- (AAC:1) (TTA:1)

**GAACAACTTA**

- (GAA:1) (AAC:1)
- (AAC:1) (ACT:1)
- (ACA:1) (CTT:1)
- (CAA:1) (TTA:1)

**TTTAGGCAAC**

- (TTT:1) (GGC:1)
- (TTA:1) (GCA:1)
- (TAG:1) (CAA:1)
- (AGG:1) (AAC:1)

---

**Map**

- ACA:1
- ATG:1
- CAA:1
- GCA:1
- TGA:1
- TTA:3

**Shuffle**

- ACT:1
- AGG:1
- CCT:1
- GGC:1
- TTT:1

**Reduce**

- AAC:3
- ACC:1
- CTT:2
- GAA:2
- TAG:1