The Value of Mate-pairs for Repeat Resolution
An Analysis on Graphs Created From Short Reads

Joshua Wetzel

Department of Computer Science
Rutgers University–Camden
in conjunction with
CBCB at University of Maryland
wetzeljo@camden.rutgers.edu

August 7th, 2009
Approx. 25 to 250 bp per read

**Advantages:**
- high throughput
- relatively low cost

**Disadvantages:**
- less certainty with error detection
- greater difficulty with resolution of repeats
  - lower likelihood of spanning an entire repeat within one read
Choose a value \( k \)

1. Node for each length \( k - 1 \) substring that exists in any read

Length \( k \) substrings are represented by directed edges between \( k - 2 \) suffix/prefix overlap

- Edge multiplicity rep. # of times a substring appears

Example

\[
\begin{align*}
&\text{CGATATTCGCTAATTCGCG} \\
&\text{ATTC} \quad \text{TTCG}
\end{align*}
\]

\[ k = 5 \]

*Pevzner et al.*
Eulerian Paths

- Every length \( k \) substring of read \( \rightarrow \) 1 edge
- With perfect data, graph is Eulerian
- Any Eulerian tour \( \rightarrow \) valid assembly of the reads

**Good News:**
- Eulerian Tour can be found quickly

**Bad News:**
- Repeats \( \rightarrow \) many Eulerian tours
Eulerian Simplifications

Three types of forks:

- Forward Fork
- Backward Fork
- Full Fork

Reduce Unambiguous Traversal Regions into Single Nodes

- Simple Path Compression
- Compressing Tree-like Regions
- Splitting Half-decision Nodes

Kingsford
A graph of only full-fork nodes (any two of which may be separated by a non-decision node)

We need additional information for finishing
Best known avenue: Use matepairs
Mate-pairs

Random fragment with a approximately known size

Both ends are sequenced

Unique path of same approximate size in the assembly graph → partial traversal of the graph

Widely accepted empirically, but no work has been done to assess their theoretical usefulness with short reads
Before We Spend Our Money

Mate-pair libraries are expensive. How much are they worth?

Can Repeats Be Resolved By Mate-pairs If Data Is Perfect?

- No sequencing errors
- Perfect coverage of the genome - *exactly one* edge in the graph for every length $k$ substring of the genome
- Assume we know *exact* length of each insert
The Plan

1. Idealized Data
2. Compress
3. Apply Matepair Info
4. Finishing Complexity Reduction

- Apply Eulerian Simplifications

Create Perfect Graphs with \( k \) as: 35, 50, 100

Simulate Various Lengths

Standard vs. Ideal

Measure
Simulating Mate-pairs

1. Choose an insert size, $j$
2. Choose a random starting point, $s$, in the known sequence
3. Choose an ending point based on a Gaussian dist. using $j$ as the mean and a 10\% stand. dev.
4. First and last $k$ letters are pairs with known distance
Reducing Finishing Complexity Using Mate-pairs

Use a shortest path heuristic
- Find a pair of nodes where the mate sequences are located
- If shortest path = insert size → assume path is correct

We can verify correctness by comparison to known sequence

An assembler cannot do this

Solution:
Only use mates if shortest path is unique
Which Mate Pairs Can Be Used?

**Good**

R1

10

10

10

R2

10

12

**Bad**

R1

10

10

R2

10

10

VS.
Goal: Reduce manual finishing efforts

Define Finishing Complexity:
- A value proportional to # of experiments needed to finish the genome
- Fin. Comp. of one node is $\sum_{i=2}^{d} i = \frac{d^2 + d}{2} - 1$

$\text{FinComplex}(G) = \text{Sum over all nodes}$
Increasing Clone Coverage to Infinity

Bartonella quintana Toulouse: Genome Size $\approx 1.5$ Mb

![Graph showing Num. Inserts vs. % Reduction in Finishing Complexity](image-url)
An Important Observation

Many areas of these Graphs Look Like This

These repeats can only be resolved by inserts that span exactly 1 repeat
What Causes Localized Complexity?

A While Ago...

Evolution

Now

1 Big Repeat

2 Small Repeats

How often does this situation arise in the graph?

Joshua Wetzel
Goal: Measure the amount of localized complexity in the graph (C-Statistic)

Label each fork in the graph as either trivial or non-trivial

Trivial

Non-Trivial
Graph Complexity Statistic

- Let $S$ be the set of all forks in the graph, $G$
- Let $N$ be the set of all non-trivial forks in $G$
- Let $C_v$ be the finishing complexity of node $v$

$$cstat(G) = \frac{\sum_{v \in N} C_v}{\sum_{v \in S} C_v} \times 100$$

**Hypothesis:** Graphs with a high C-Stat will not experience good reductions in finishing complexity using traditional mate-pair libraries
Choosing Graph-Specific Insert Sizes is Important
Tailoring Mate-pair Size to the Graph

Typical sequencing experiments choose 2 common size mate-pair libraries without considering repeat size

- Typical sizes: 2000, 6000, 8000, 35000 (bp)

**Hypothesis:**
Given the prevalence of non-trivial repeats, we can likely reduce finishing complexity more by choosing 2 library sizes targeted at just barely crossing the forks of greatest complexity

How do we choose what size inserts to use?
Complexity Bar Chart for Thermus thermophilus

Finish Complexity

Repeat Length Range (bp)

(0, 200) (201, 401) (402, 602) (804, 1004) (1005, 1205) (1206, 1406)
How the Complexity Bar Chart is Used

Two tallest bins (most combined complexity)

For of these two bins:
1. Take the mean repeat length of all the repeats in the bin
2. Add $3 \times k$ to the avg.
3. Create inserts of this approximate size

Average optimal mate-pair length to cover nodes of most complexity was between $4.5k$ and $6k$

- Small $\sigma$ for graphs with C-Stat $\geq 50$
- Large $\sigma$ for graphs with C-Stat $< 50$
Want mate-pair to begin in a non-decision node and cross exactly one repeat

In worst case non-decision node maybe be as short as $k$

Want at least one insert for every $k$ bps in the genome

If average arrival rate, $\lambda$, is 10 across $k$ bp

$$P[X = x] = \frac{\lambda^x e^{-\lambda}}{x!}$$

$$P[X = 1] = \frac{10}{e^{10}} \approx 0.00045$$

$$P[X < 1] \leq 0.00045$$

To get $\lambda = 10$ across $k$ bp we need $10 \frac{G}{k}$ inserts
Simulations Across 360 Organisms

How Much Can We Reduce Finishing Complexity Using 2 libraries each with $10 \frac{G}{K}$ # of inserts if we use:

- 2 standard mate-pair libraries
- The 2 'ideal' libraries based on the complexity chart
Using Two ’Ideal’ Mate-Pair Libraries (Avgs: 4.5k to 6k for smallest)

Overall Avg Reduction: 82.70%, $\sigma \approx 17\%$
Overall Avg Reduction: 47.52%, $\sigma \approx 22\%$
Local Complexity Vs. Global Complexity (Using 2000 and 8000)

Total Finishing Complexity vs. C–Statistic

% Reduction in Finishing Complexity
- < 80%
- >=80%

Joshua Wetzel
Some Conclusions

- **Short inserts are better for repeat resolution**

- A large confounding factor in repeat resolution is **localized complexity** caused by intra-repeat nucleotide mutations

- The **C-Stat** is simple and useful measurement of localized complexity
  
  - Most graphs have a C-Stat $\geq 60$
  
  - If C-Stat $\geq 50$, then standard mp libs. are not useful for resolving repeats

- **Tailoring mp libs. to the repeat structure** of the assembly graph is a useful technique
Even with perfect data and an ideal approach to creating mate-pair libs, we can only resolve an avg. of 83% of finishing complexity ($\sigma \approx 17\%$).

Does not bode well for noisy data.

On the other hand:

The only known technique for better resolving localized complexity is use of longer reads.
Thanks