

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, ...	
It was	the	best	of	times,	it	was	the	worst	of	times,	it	was	the	age	of	wisdom,	it	was	the	age	of	foolishness, ...	
It was	the	best	of	times,	it	was	the	worst	of	times,	it	was	the	age	of	wisdom,	it	was	the	age	of	foolishness, ...	
It was	the	best	of	times,	it	was	the	worst	of	times,	it	was	the	age	of	wisdom,	it	was	the	age	of	foolishness, ...	
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 x 138,656 words / 5 words per fragment = 138k fragments
 - The short fragments from every copy are mixed together
 - Some fragments are identical

Greedy Reconstruction

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

It was the best of
was the best of times,
the best of times, it
best of times, it was
of times, it was the
of times, it was the
times, it was the worst
times, it was the age

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

Original Fragment

It was the best of

Directed Edge

It was the best → was the best of

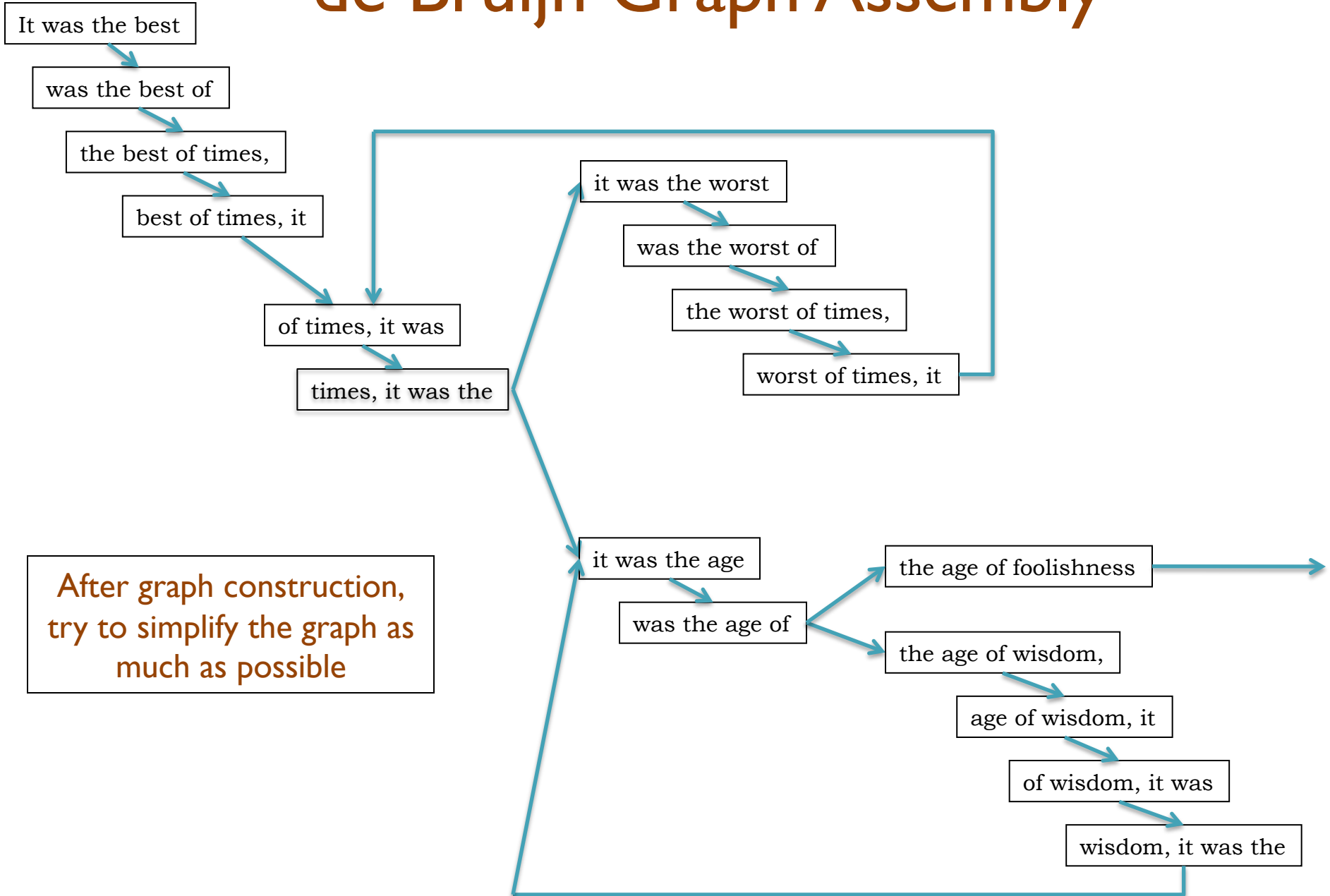
- Locally constructed graph reveals the global sequence structure
 - Overlaps between sequences implicitly computed

de Bruijn, 1946

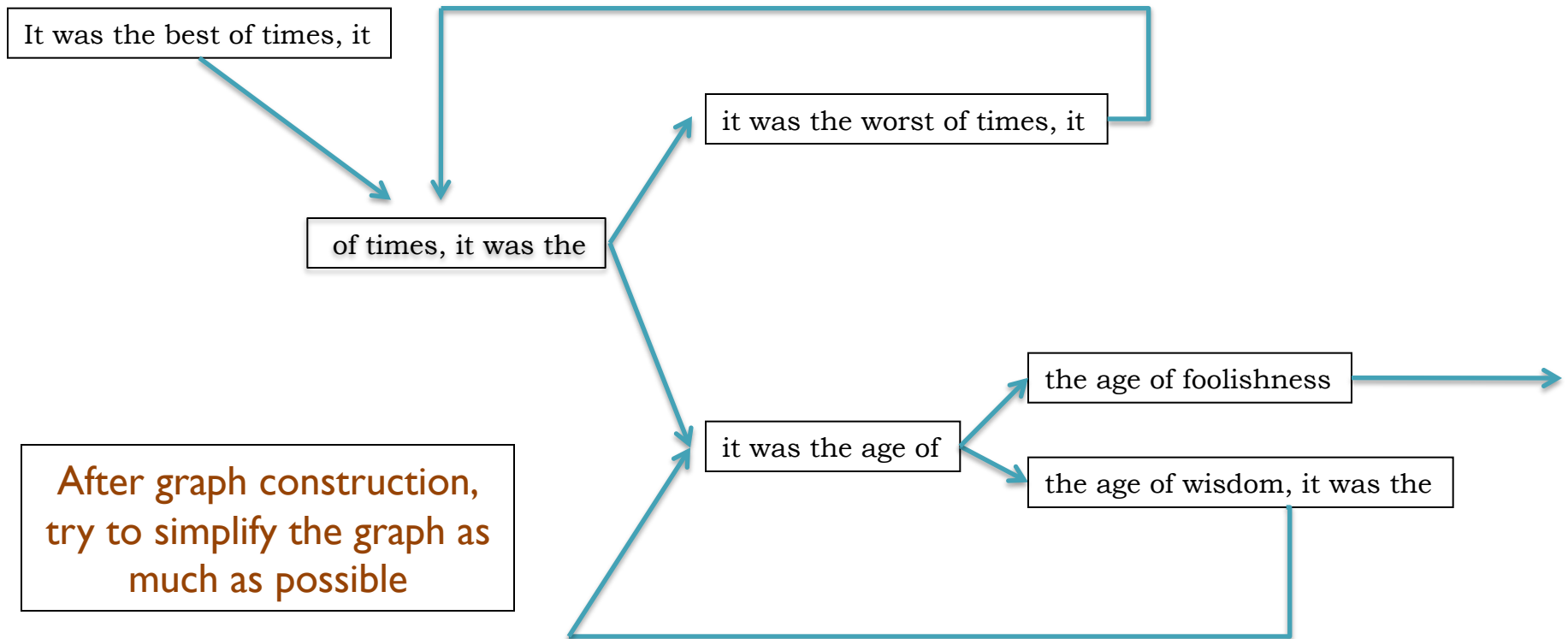
Idury and Waterman, 1995

Pevzner, Tang, Waterman, 2001

de Bruijn Graph Assembly



de Bruijn Graph Assembly

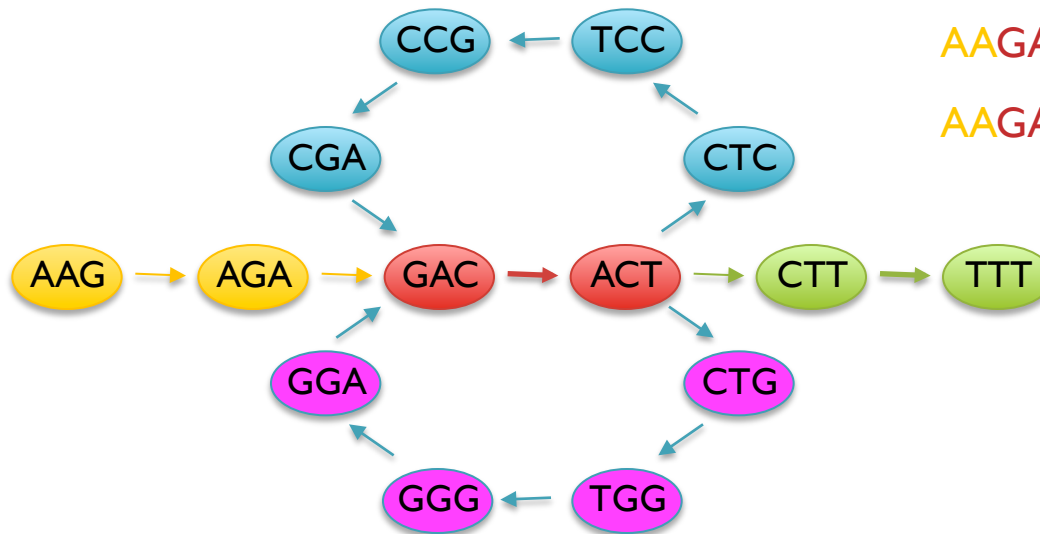


Short Read Assembly

Reads

AAGA
ACTT
ACTC
ACTG
AGAG
CCGA
CGAC
CTCC
CTGG
CTTT
...

de Bruijn Graph



Potential Genomes

AAGACTCCGACTGGGACTTT

AAGACTGGGACTCCGACTTT

- 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
 - ABySS (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

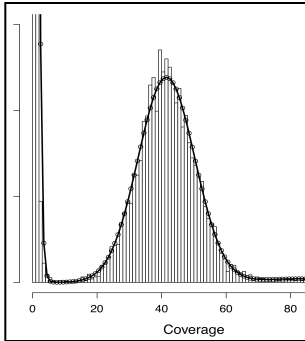
<http://hadoop.apache.org>

- 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

<http://www.cbcb.umd.edu/software/quake/>

(Kelley, Schatz, Salzberg, 2010*)

CloudBurst



Highly Sensitive Short Read Mapping with MapReduce

100x speedup mapping on 96 cores @ Amazon

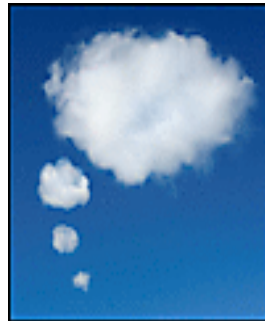
<http://cloudburst-bio.sf.net>

(Schatz, 2009)

Crossbow

Searching for SNPs with Cloud Computing

SNPs in 4 hours and ~\$85 per human genome



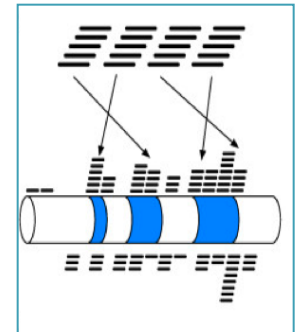
(Langmead, Schatz, Lin, Pop, Salzberg, 2009)

<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

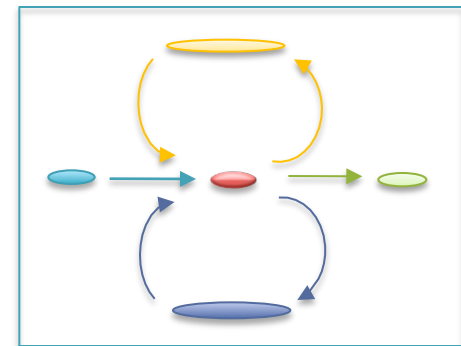
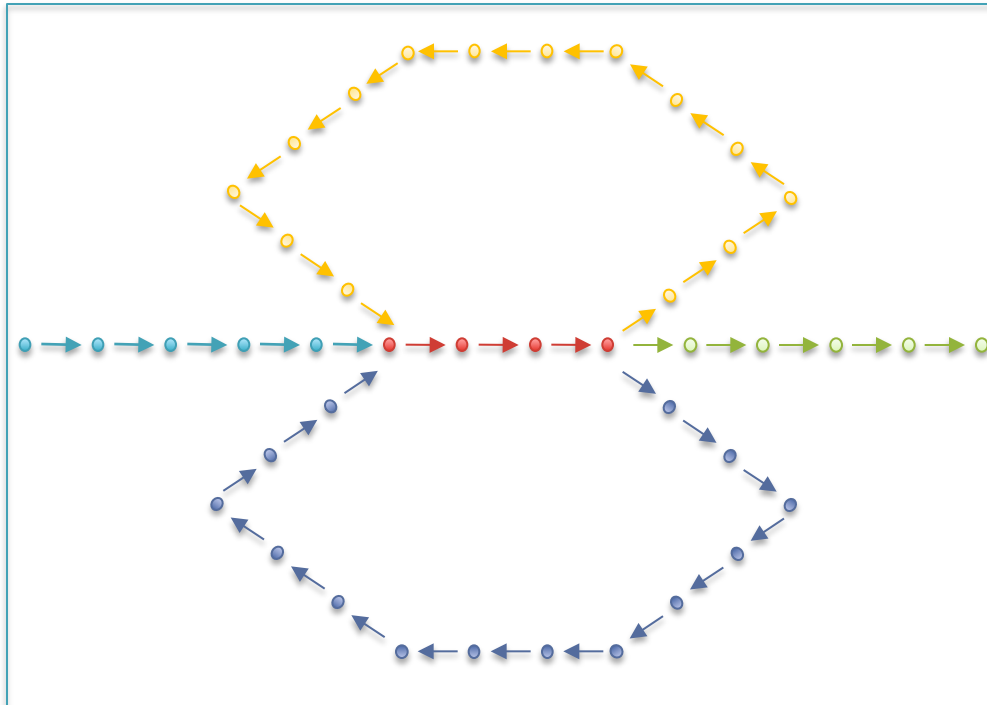


(Langmead, Hansen, Leek, 2010)

<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.

Lin, J., Schatz, M.C. (2010) Workshop on Mining and Learning with Graphs Workshop (MLG-2010)

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

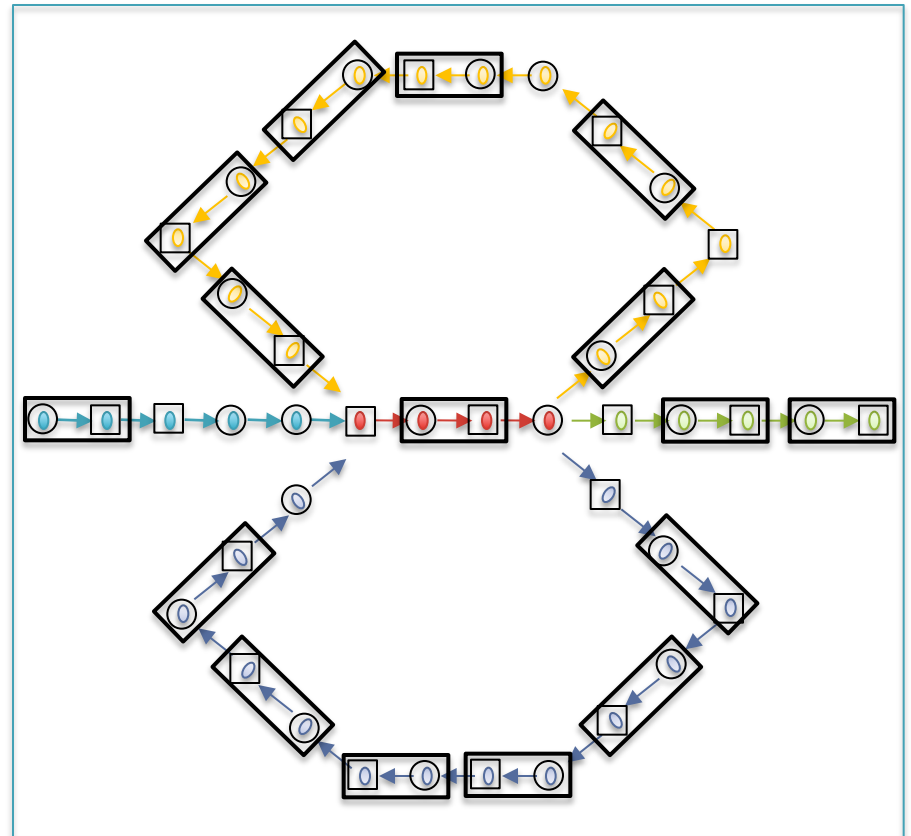
Fast Path Compression

Challenges

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

Randomized List Ranking

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



Initial Graph: 42 nodes

Randomized Speed-ups in Parallel Computation.

Vishkin U. (1984) *ACM Symposium on Theory of Computation*. 230-239.

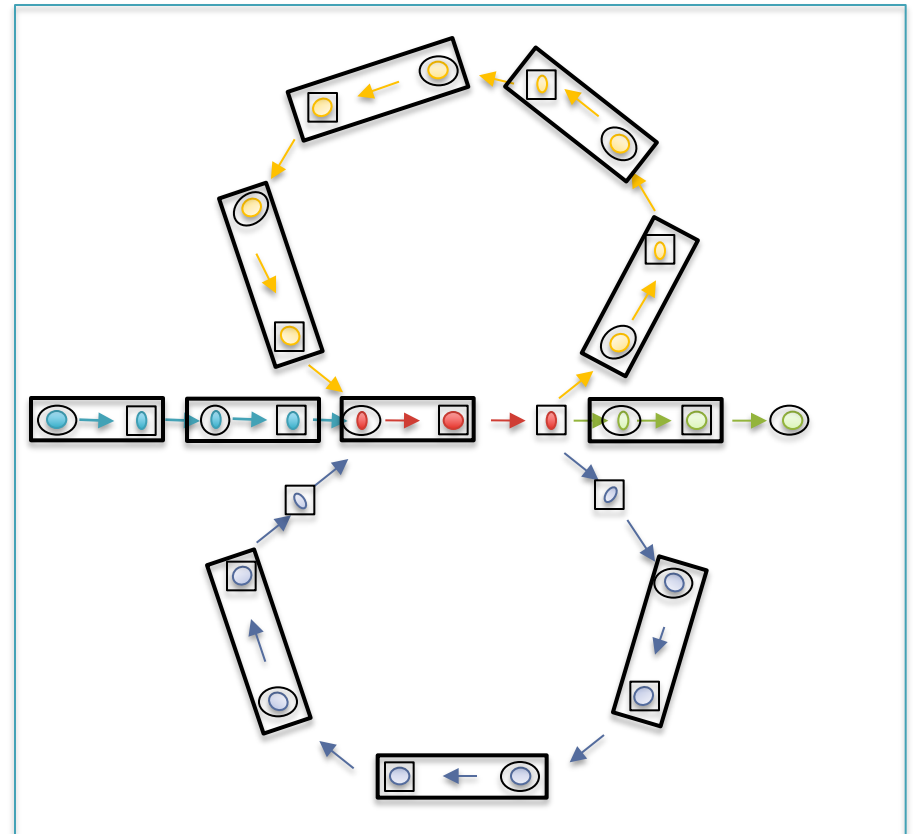
Fast Path Compression

Challenges

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

Randomized List Ranking

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



Round 1: 26 nodes (38% savings)

Randomized Speed-ups in Parallel Computation.

Vishkin U. (1984) *ACM Symposium on Theory of Computation*. 230-239.

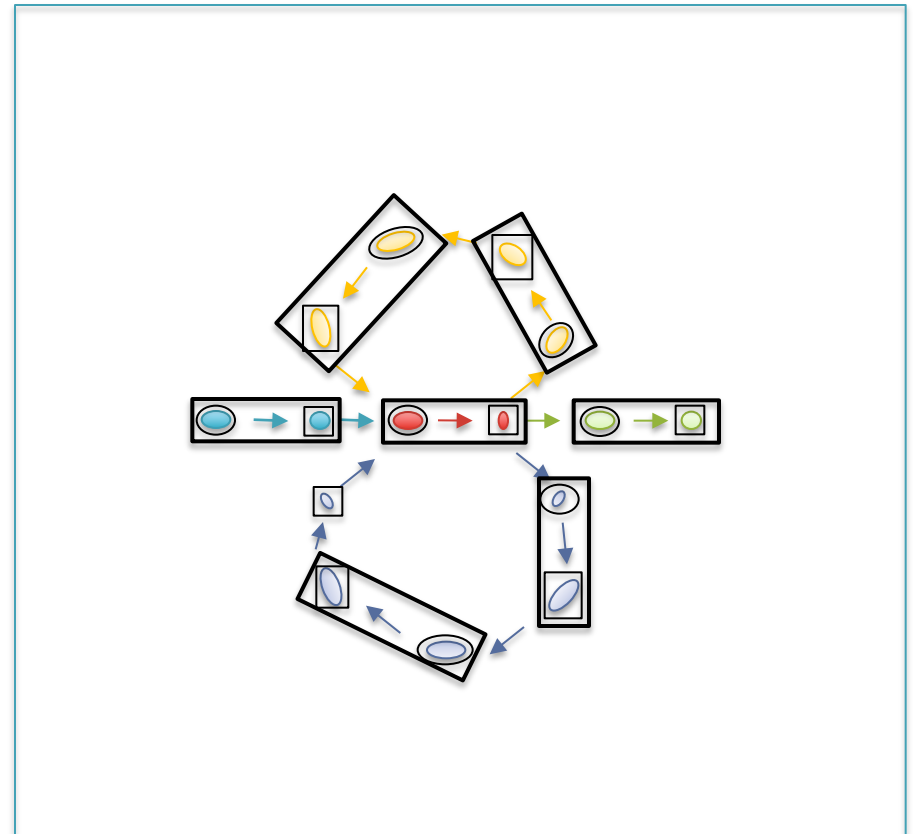
Fast Path Compression

Challenges

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

Randomized List Ranking

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



Round 2: 15 nodes (64% savings)

Randomized Speed-ups in Parallel Computation.

Vishkin U. (1984) *ACM Symposium on Theory of Computation*. 230-239.

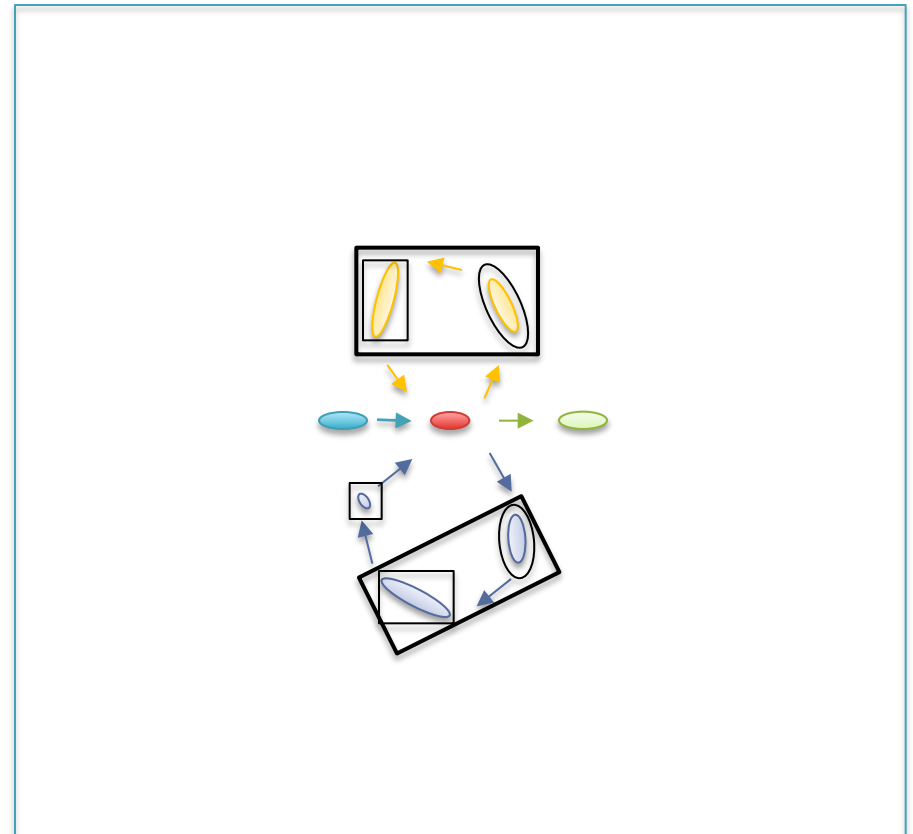
Fast Path Compression

Challenges

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

Randomized List Ranking

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



Round 2: 8 nodes (81% savings)

Randomized Speed-ups in Parallel Computation.

Vishkin U. (1984) *ACM Symposium on Theory of Computation*. 230-239.

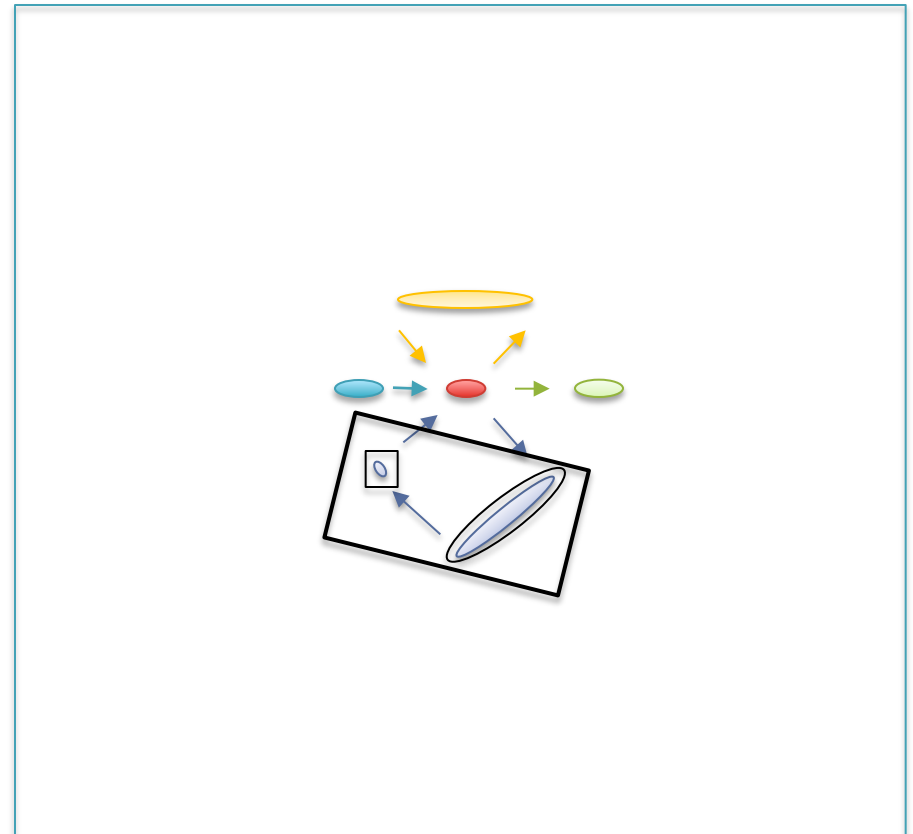
Fast Path Compression

Challenges

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

Randomized List Ranking

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



Round 3: 6 nodes (86% savings)

Randomized Speed-ups in Parallel Computation.

Vishkin U. (1984) *ACM Symposium on Theory of Computation*. 230-239.

Fast Path Compression

Challenges

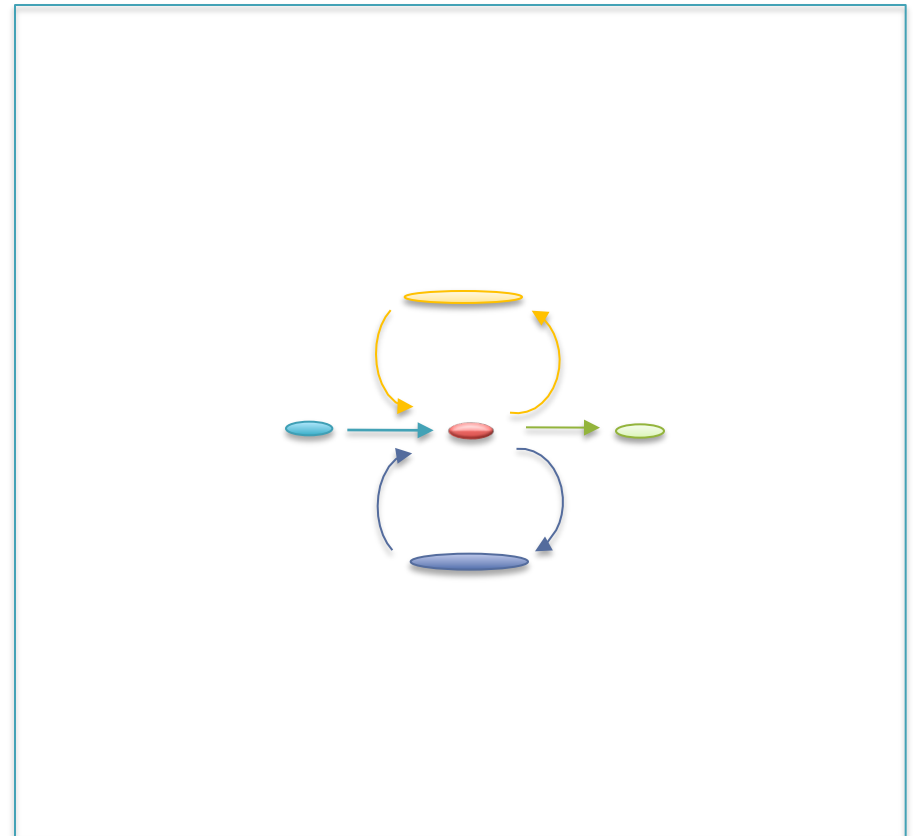
- Nodes stored on different computers
- Nodes can only access direct neighbors
- No "Tournament Bracket"

Randomized List Ranking

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

Performance

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

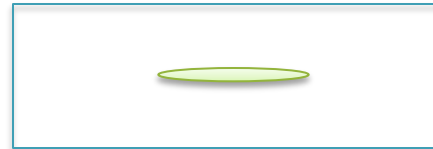
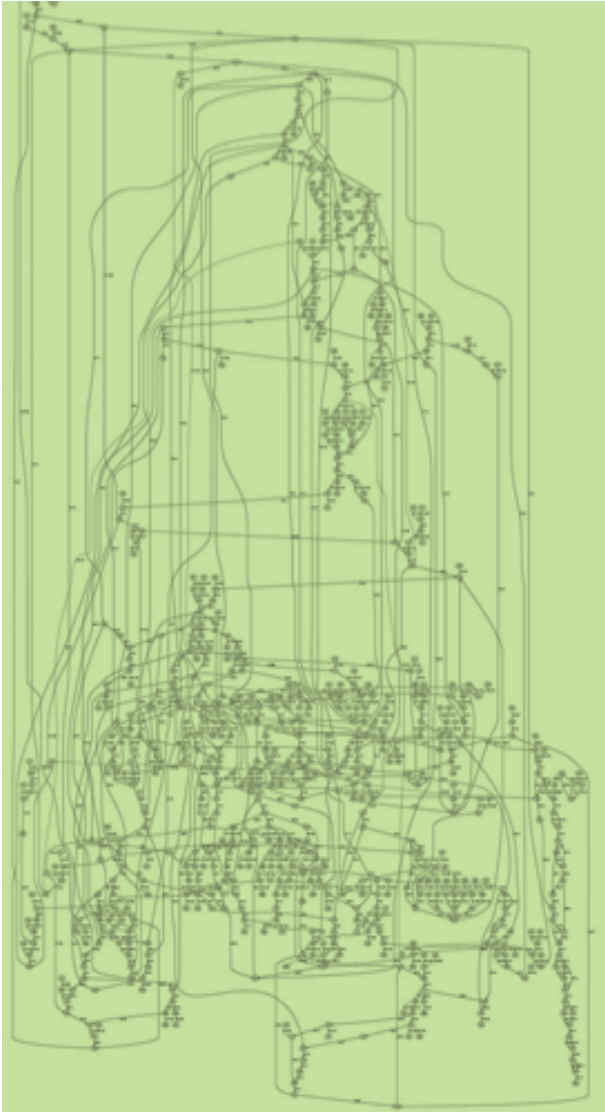


Round 4: 5 nodes (88% savings)

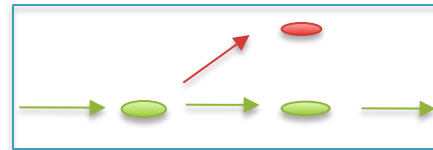
Randomized Speed-ups in Parallel Computation.

Vishkin U. (1984) *ACM Symposium on Theory of Computation*. 230-239.

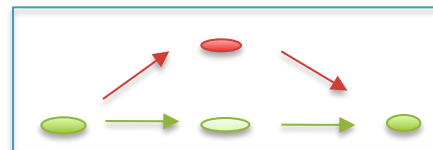
Node Types



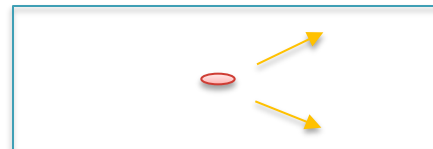
Isolated nodes (10%)



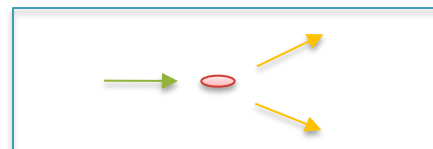
Tips (46%)



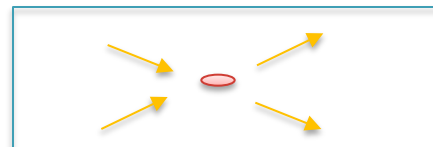
Bubbles/Non-branch (9%)



Dead Ends (.2%)



Half Branch (25%)



Full Branch (10%)

(Chaisson, 2009)

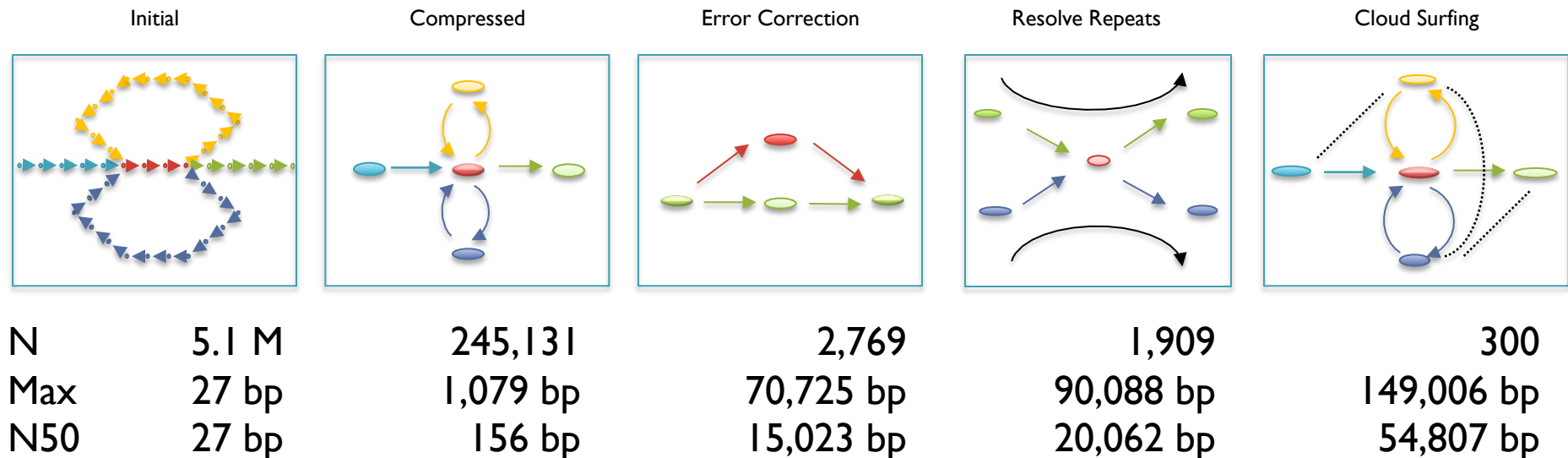
Contrail

<http://contrail-bio.sourceforge.net>



De novo bacterial assembly

- *Genome: E. coli* K12 MGI655, 4.6Mbp
- *Input: 20.8M* 36bp reads, 200bp insert (~150x coverage)
- *Preprocessor: Quake* Error Correction



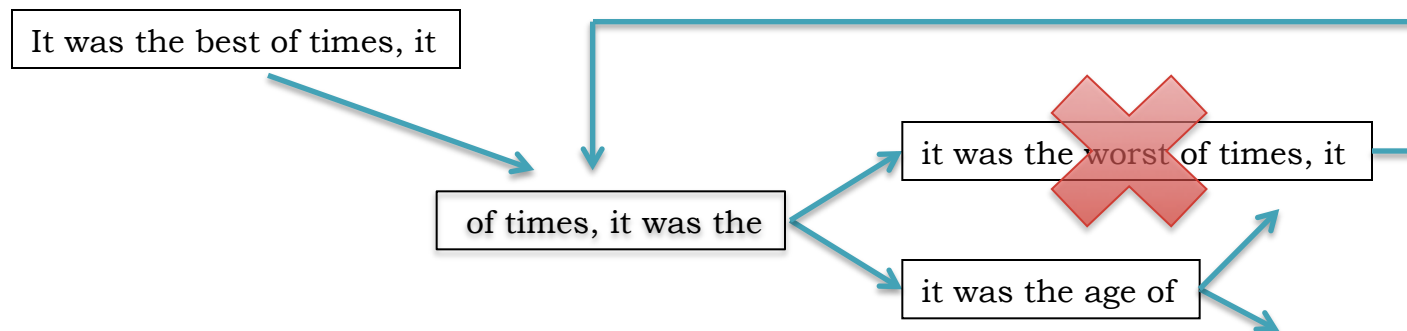
Assembly of Large Genomes with Cloud Computing.

Schatz MC, Sommer D, Kelley D, Pop M, et al. *In Preparation.*

E. coli Assembly Quality

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

Assembler	Contigs \geq 100bp	N50 (bp)	Incorrect contigs
Contrail PE	300	54,807	4
Contrail SE	529	20,062	0
SOAPdenovo PE	182	89,000	5
ABYSS PE	233	45,362	13
Velvet PE	286	54,459	9
EULER-SR PE	216	57,497	26
SSAKE SE	931	11,450	38
Edena SE	680	16,430	6



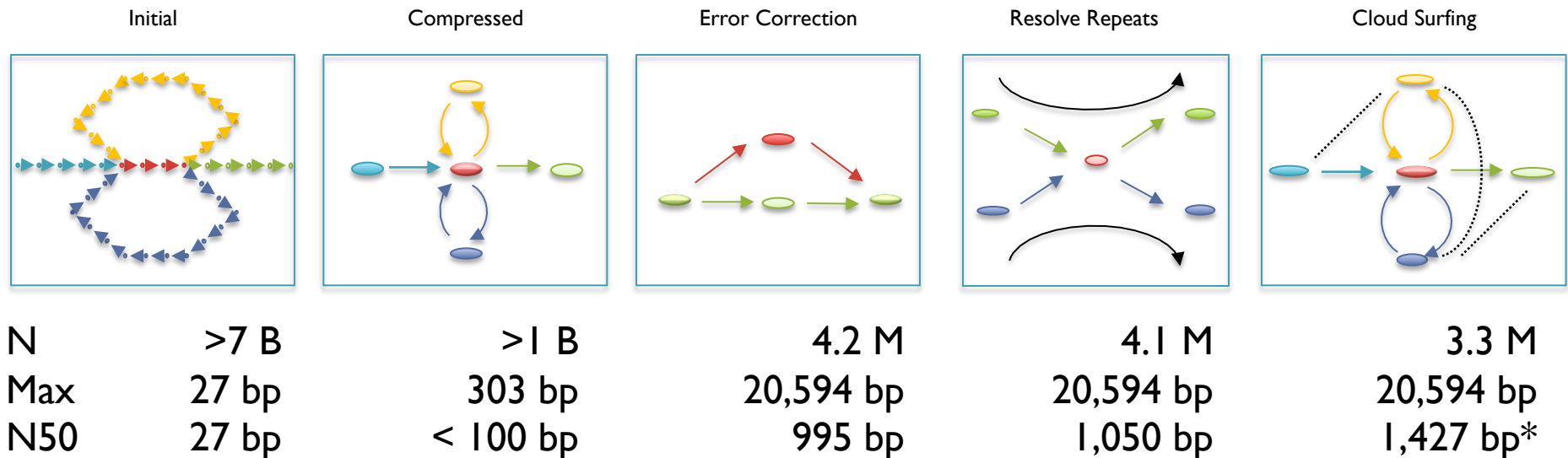
Contrail

<http://contrail-bio.sourceforge.net>



De novo Assembly of the Human Genome

- *Genome*: African male NAI8507 (SRA000271, Bentley *et al.*, 2008)
- *Input*: 3.5B 36bp reads, 210bp insert (~40x coverage)



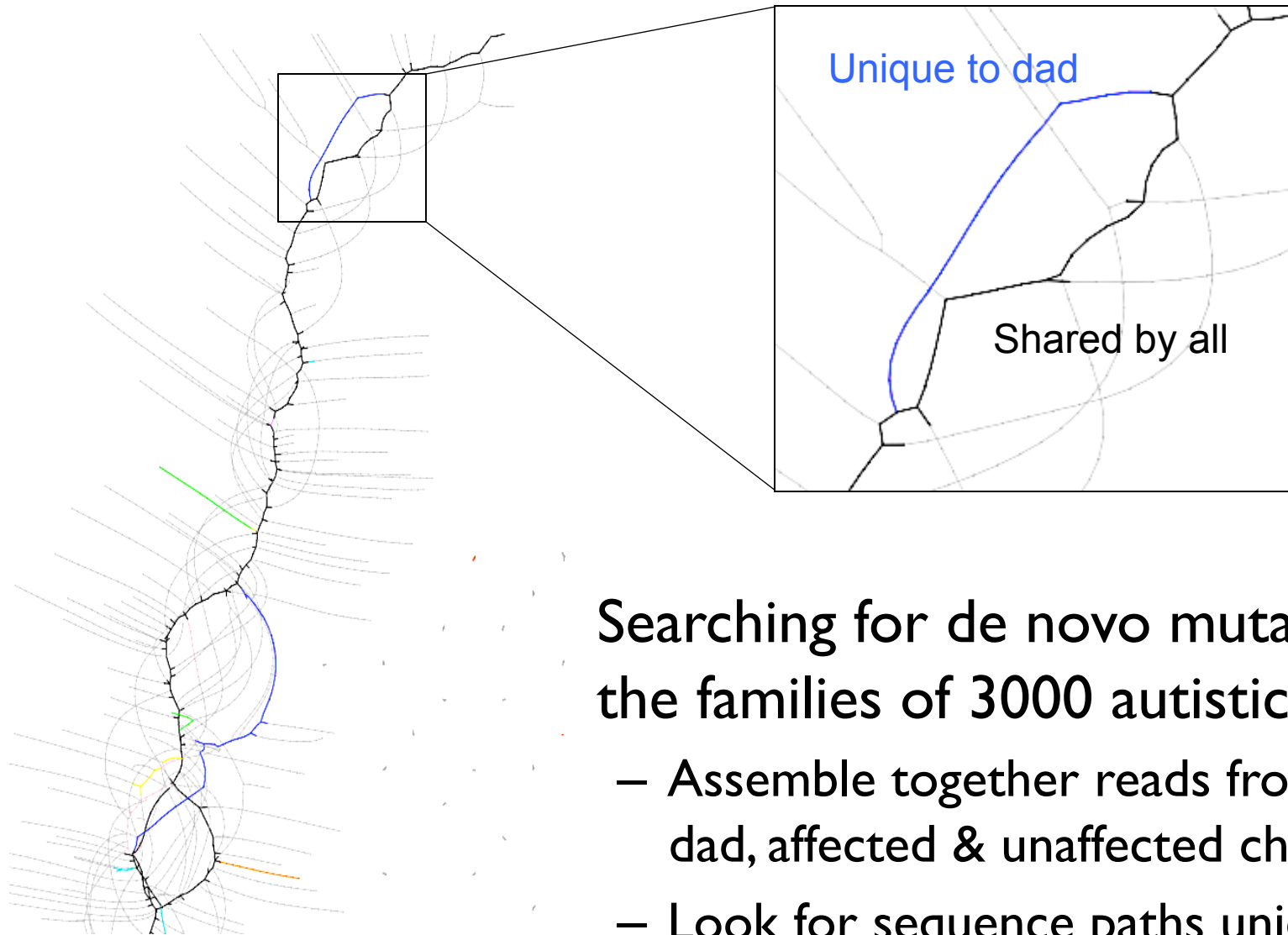
Assembly of Large Genomes with Cloud Computing.

Schatz MC, Sommer D, Kelley D, Pop M, *et al.* *In Preparation.*

One more thing...



Variations and de Bruijn Graphs



MRCILI

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

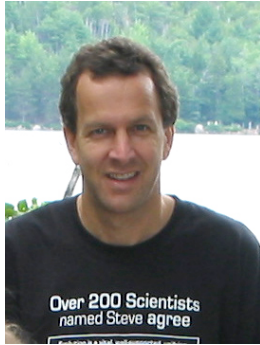
- Assemble together reads from mom, dad, affected & unaffected children
- Look for sequence paths unique to affected child



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 → key:value pairs
 - **Shuffle**: Group together pairs with same key
 - **Reduce**: key, value-lists → output

Map, Shuffle & Reduce
All Run in Parallel

