Commodity computing in genomics research

Mihai Pop
Mike Schatz
Dan Sommer

Department of Computer Science
Center for Bioinformatics and Computational Biology
University of Maryland
College Park

UMIACS
University of Maryland Institute for Advanced Computer Studies

CBCB
Center for Bioinformatics & Computational Biology

University of Maryland

Facing a deluge of biological data

- DNA sequencing – by 2012 ~ Petabytes/year
  - more than the Hadron collider (Flicek, Genom. Biol. 2009)
  - unlike physics – large installed base of instruments generating data
  - personal genomics (1000 Genome project)
  - human microbiome project
  - environmental metagenomics
- Bio-medical imaging
  - better microscopes
- Other high-throughput technologies
  - mapping
  - phenotyping
  - etc...

We do not know how to:

store
transfer
analyze

these data-sets efficiently

The evolution of DNA sequencing

Since	Technology	Read length	Throughput/run	Throughput/hour	cost/run
1977-	Sanger sequencing	> 1000bp	4hr 400-500 kbp	100 kbp *25 kB*	$200
2005-	454 pyrosequencing	250-400bp	4hr 100-500 Mbp	25-100 Mbp *6-25 MB*	$13,000
2006-	Illumina/Solexa	50-100bp	3 days 2-3 Gbp	25-40 Mbp *6-10 MB*	$3,000
2007-	ABI SOLiD	35-50bp	3 days 6-20 Gbp	75-250 Mbp *19-60 MB*	est. $3-5,000
2008-	Helicos **single molecule**	25-50 bp	8 days 10 Gbp	~50 Mbp est. 1Gbp/hour *250 MB*	~$18,000
TBA (2010)	Pacific Biosciences **single molecule**	100-200 kbp	?	?	?

Helicos - ~500-600 kbps throughput in just DNA letters (usually a lot more info produced)
DVD ~ 8Mbps, BlueRay ~40Mbps

Can cloud computing help?

PROS

- Ease of programming
  - – many biotech programmers do not have formal CS training
  - – MapReduce may be "simple" enough
  - – currently working with undergrad interns
- Can existing software be adapted to a parallel setting?
  - –YES (stay tuned)
- Cost structure
  - – computation as "lab consumable" instead of "infrastructure"

Can cloud computing help?...cont

CONS/CHALLENGES

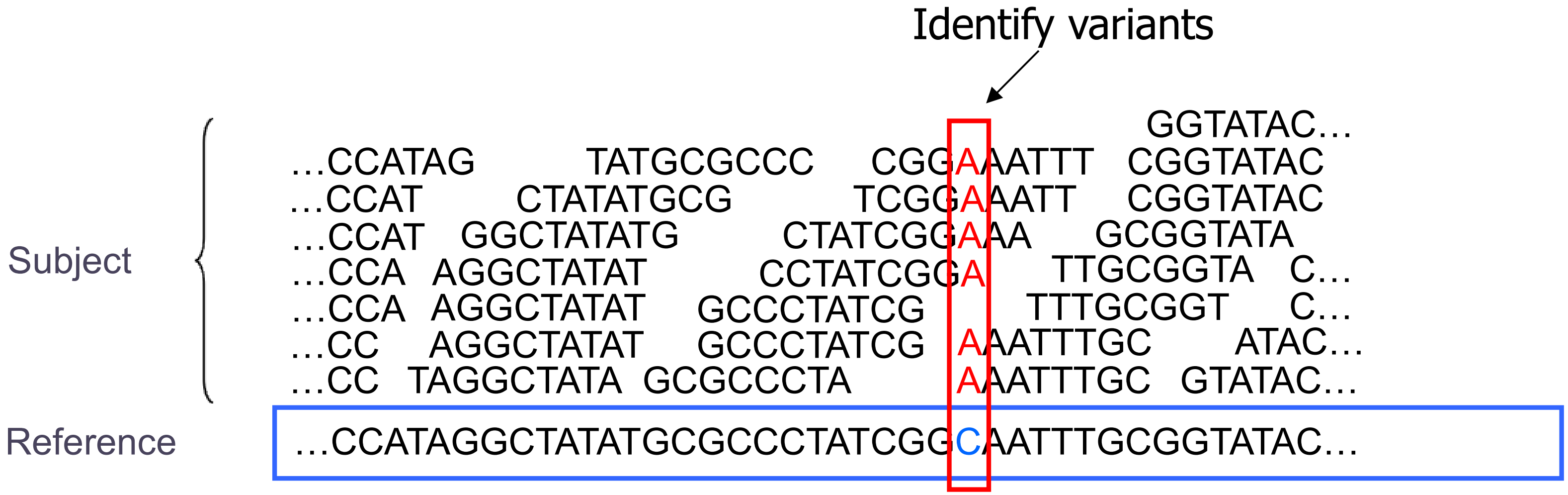
- Communication costs (local vs. remote cluster)
- Data privacy/security (HIPAA)

What bioinformatics tools work in the "cloud"?

- Various sequence alignment (string matching) tasks
  - "embarassingly" parallel
  - already successfully handled through condor/sungrid/LSF, MPI, custom parallel hardware
  - will show: work well in MapReduce (CloudBurst)
  - actually: can adapt existing software to MapReduce (Crossbow)

- Genome assembly ("best" superstring)
  - hard to parallelize (graph algorithms)
  - for most genomes many possible solutions (> 1 google)
  - limited success demonstrated in MPI, BlueGene
  - will show: can be done in MapReduce (but tricky)
  - how well? (pending)

Short Read Mapping


Identify variants
GGTATAC…
…CCATAG TATGCGCCC CGGAAATTT CGGTATAC
…CCAT CTATATGCG TCGGAAATT CGGTATAC
…CCAT GGCTATATG CTATCGGAAA GCGGTATA
…CCA AGGCTATAT CCTATCGGA TTGCGGTA C…
…CCA AGGCTATAT GCCCTATCG TTTGCGGT C…
…CC AGGCTATAT GCCCTATCG AAATTTGC ATAC…
…CC TAGGCTATA GCGCCCTA AAATTTGC GTATAC…
Subject
Reference
…CCATAGGCTATATGCGCCCTATCGGCAATTTGCGGTATAC…

- Recent studies of entire human genomes analyzed billions of reads
  - –Asian Individual Genome: 3.3 Billion 35bp, 104 GB (Wang et al., 2008)
  - –African Individual Genome: 4.0 Billion 35bp, 144 GB (Bentley et al., 2008)

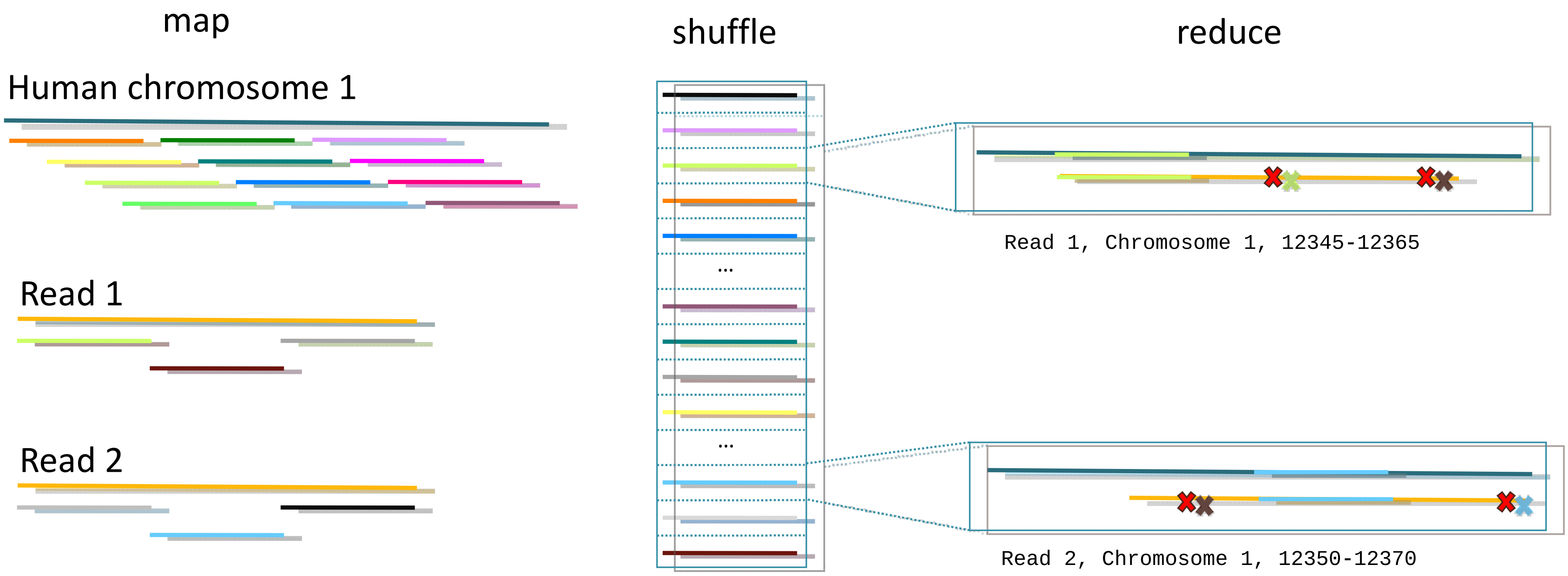
- Alignment computation required >10,000 CPU hours*
  - –Alignments are “embarassingly parallel” by read
  - –Variant detection is parallel by chromosome region

CloudBurst

1. Map: Catalog K-mers
   - Emit k-mers in the genome and reads

2. Shuffle: Collect Seeds
   - Conceptually build a hash table of k-mers and their occurrences

3. Reduce: End-to-end alignment
   - If read aligns end-to-end with ≤ k errors, record the alignment



map

shuffle

reduce

Human chromosome 1

Read 1

Read 2

…

…

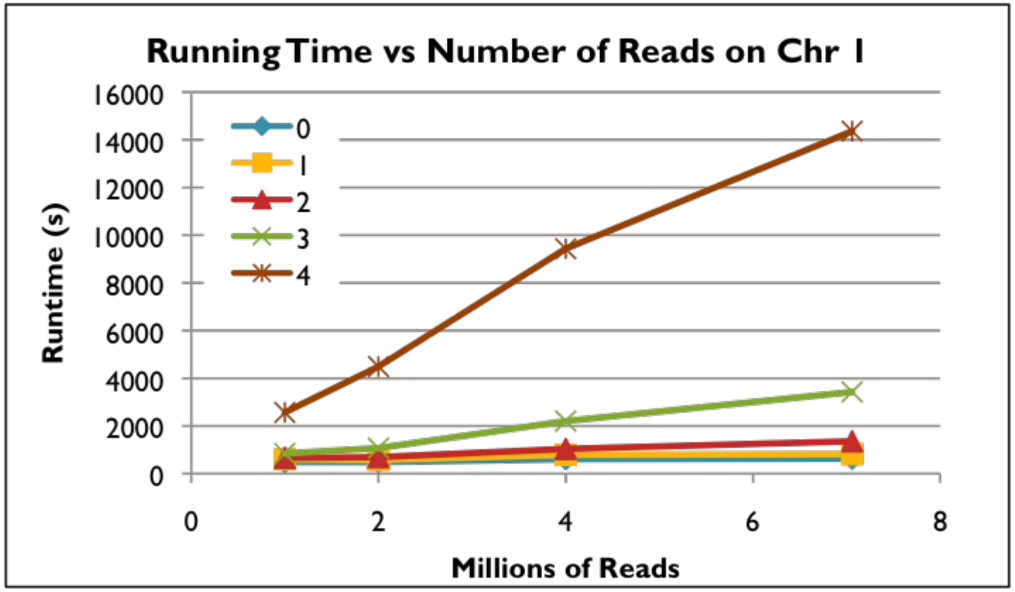
```
Read 1, Chromosome 1, 12345-12365
```

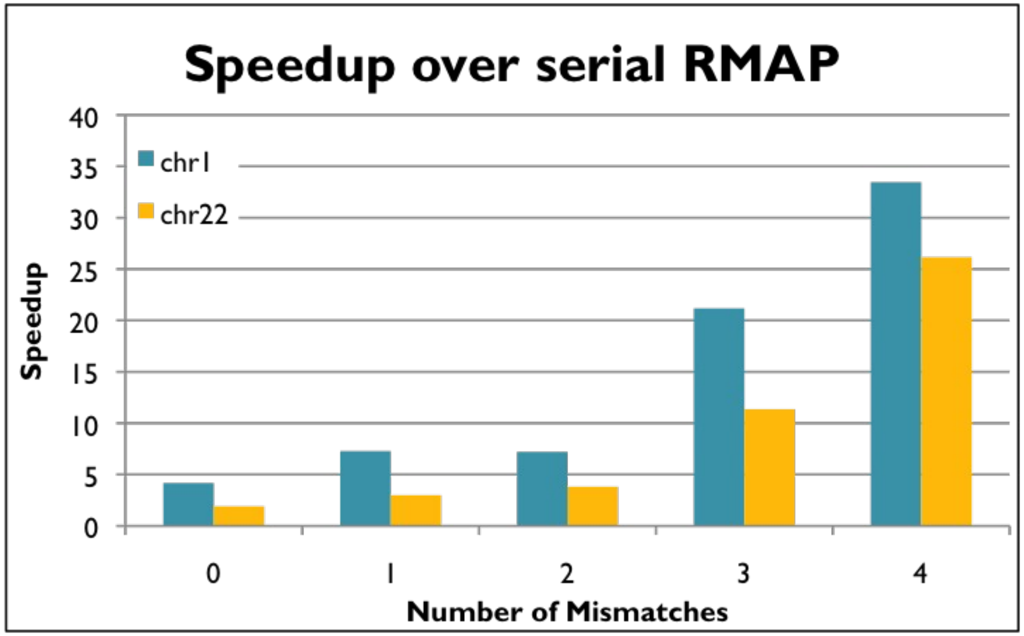
```
Read 2, Chromosome 1, 12350-12370
```

Schatz, MC (2009) CloudBurst: Highly Sensitive Read Mapping with MapReduce. *Bioinformatics*. 25:1363-1369

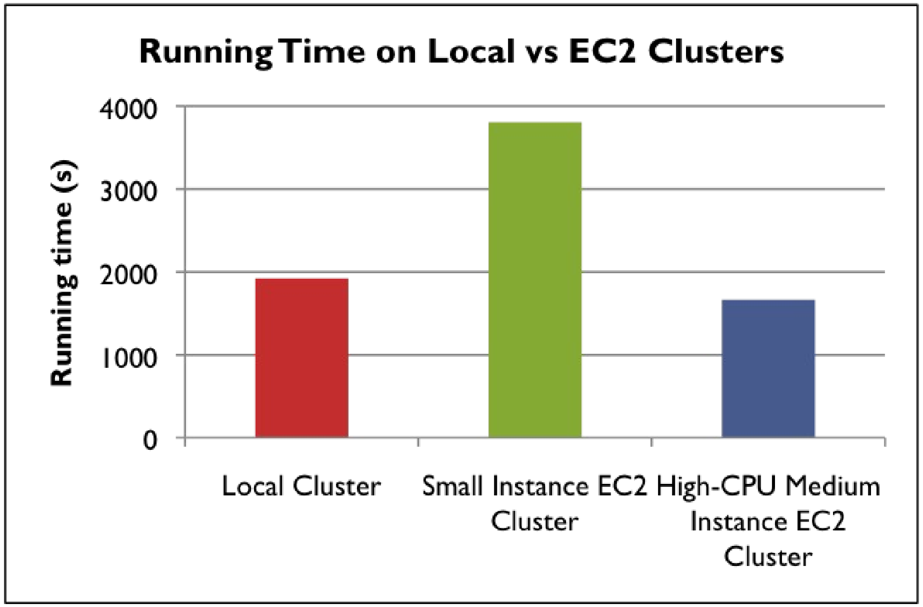
CloudBurst Results

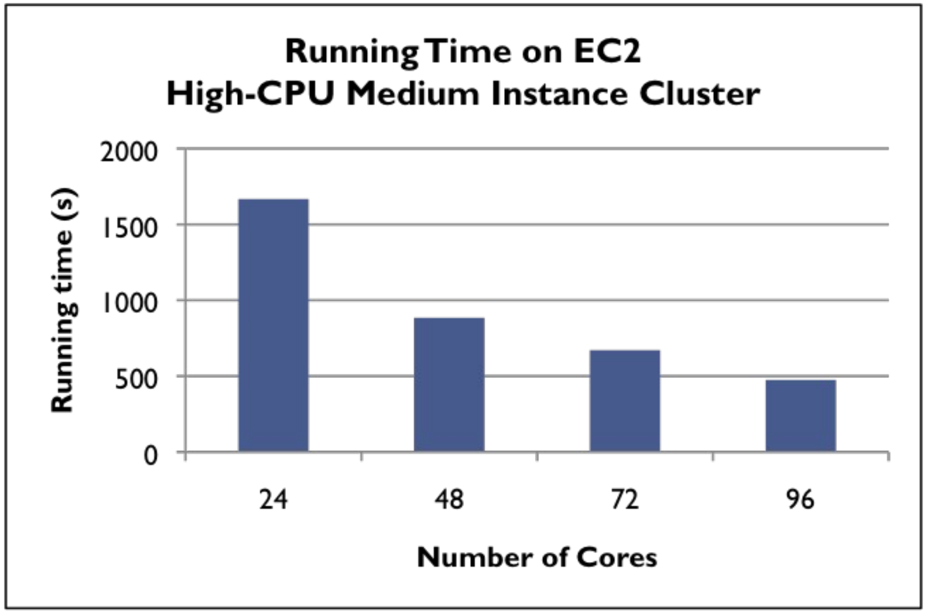
- Evaluate running time on local 24 core cluster
  - Running time increases linearly with the number of reads
- Compare to RMAP
  - Highly sensitive alignments have better than 24x linear speedup.
- Produces identical results in a fraction of the time





EC2 Evaluation





- CloudBurst running times for mapping 7M reads to human chromosome 22 with at most 4 mismatches on the local and EC 2 clusters.

- The 24-core Amazon High-CPU Medium Instance EC2 cluster is faster than the 24-core Small Instance EC2 cluster, and the 24-core local dedicated cluster.

- The 96-core cluster is 3.5x faster than the 24-core, and 100x faster than serial RMAP.

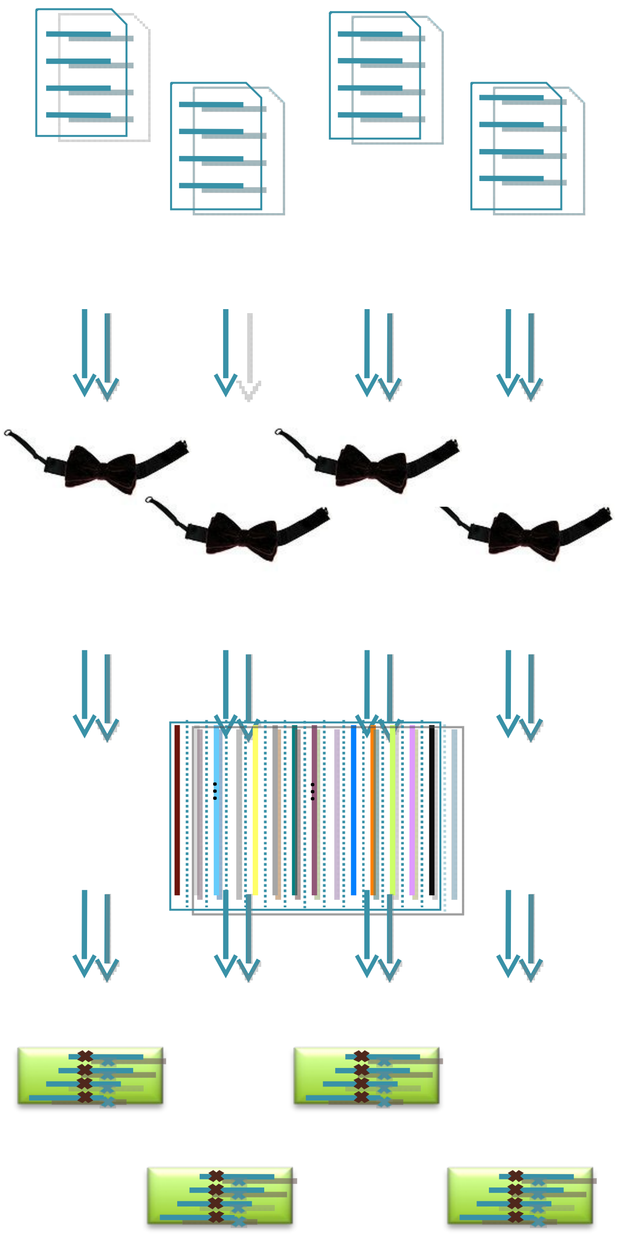
Crossbow: Rapid Whole Genome SNP Analysis

- Align billions of reads and find SNPs
  - Reuse software components: Hadoop Streaming

- Map: Bowtie
  - Emit (chromome region, alignment)

- Shuffle: Hadoop
  - Group and sort alignments by region

- Reduce: SoapSNP (Li et al, 2009)
  - Scan alignments for divergent columns
  - Accounts for sequencing error, known SNPs



Preliminary Results: Whole Genome

	Asian Individual Genome		
Data Loading	SE: 2.0 B, 24x PE: 1.3 B, 15x	106.5 GB compressed	$10.65
Data Transfer	1 hour	39+1 Small	$4.00
Preprocessing	0.5 hours	40+1 X-Large	$16.40
Alignment	1.5 hours	40+1 X-Large	$49.20
Variant Calling	1.0 hours	40+1 X-Large	$32.80
End-to-end	4 hours		$113.05

Goal: Reproduce the analysis by Wang et al. for ~$100 in an afternoon.

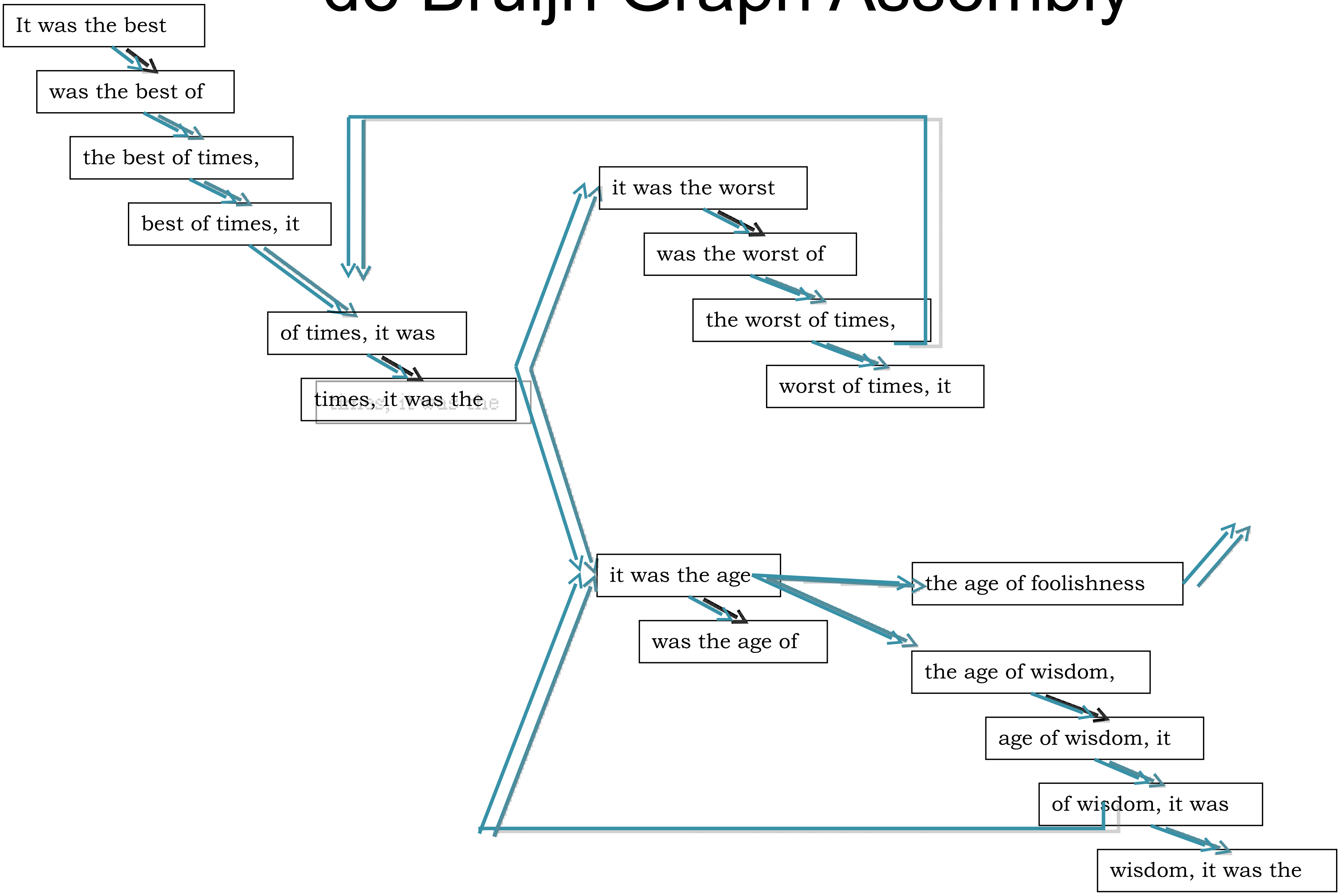
Genome assembly

- Problem: Reconstruct a genome from a collection of (imperfect) short fragments (reads)

- Two paradigms:
  - de Bruijn graph (Pevzner):
    nodes = k-mers; edges = adjacent k-mers overlap by k-1 letters
  - string/overlap graph (Myers):
    nodes = reads; edges = adjacent reads are overlapping

- Both translate into finding an Eulerian/Chinese postman path or cycle

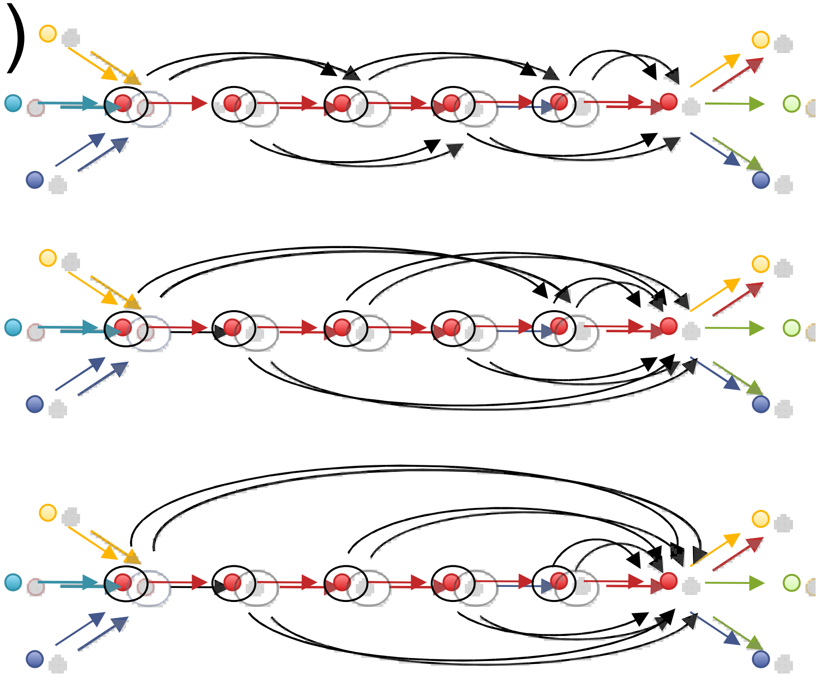
de Bruijn Graph Assembly



deBruijn assembly in the cloud

- Graph construction:
  - –Map: Scan reads and emit (ki,ki+1) for consecutive k-mers (also consider reverse complement k-mers, build bi-directed graph)
  - Reduce: Save adjacency representation of graph (n, nodeinfo)

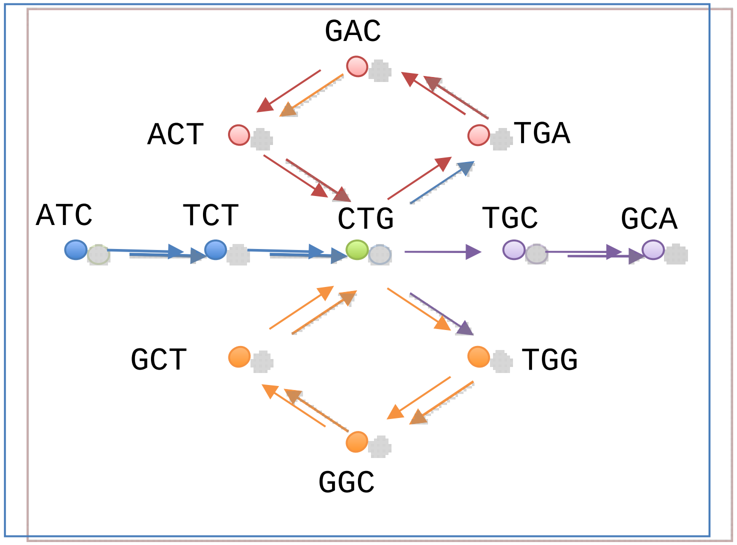
- Graph simplifications:
  - collapse simple paths (pointer jumping)
  - clean up errors (spurs & bubbles)
  - collapse trees of cycles (regions w/ unique reconstruction)



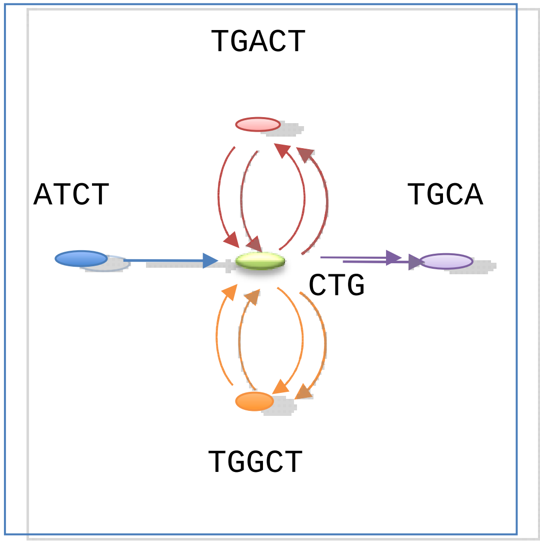
Reads

ACTG
ATCT
CTGA
CTGG
CTGC
GACT
GCTG
GGCT
TCTG
TGAC
TGCA
TGGC

Initial de Bruijn Graph



Compressed Graph



Apis mellifera genome: 236 Mbp

Reads: 24.7 M x 75bp = 1.8 Gbp
Estimated Coverage: 7.5x

Max:	774
Cnt ≥ 100:	348,440
Sum ≥ 100:	49,634,352

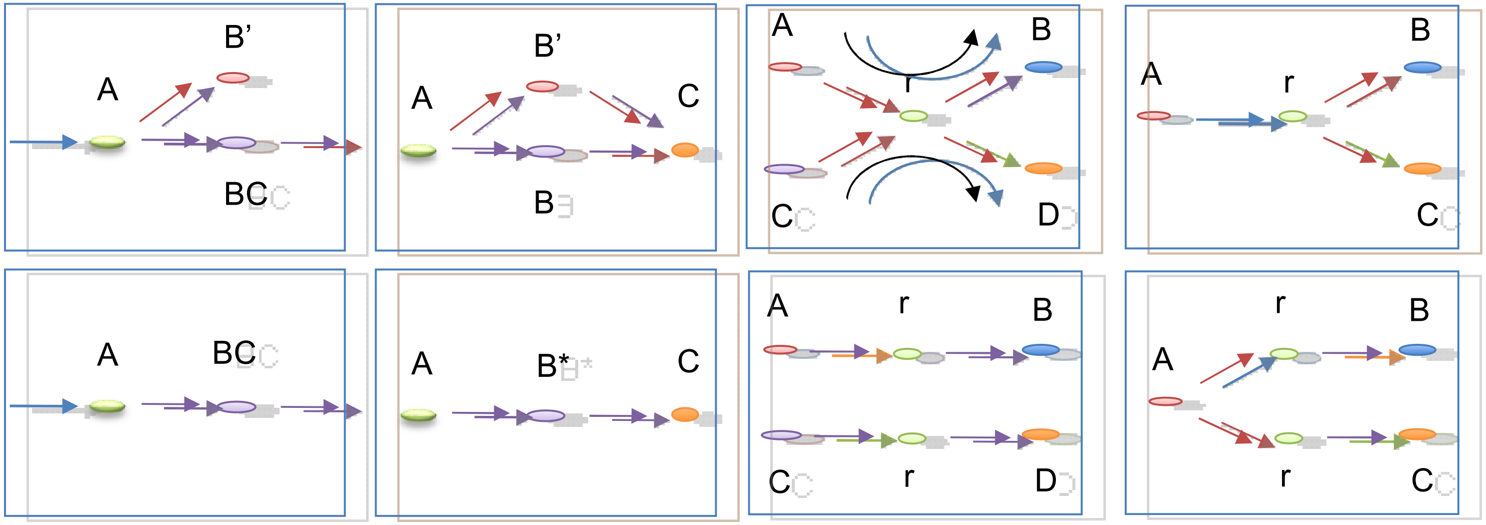
15 hours, 4GB RAM
vs
1 day, 100GB RAM

Remove Tips

Pop Bubbles

Thread Reads

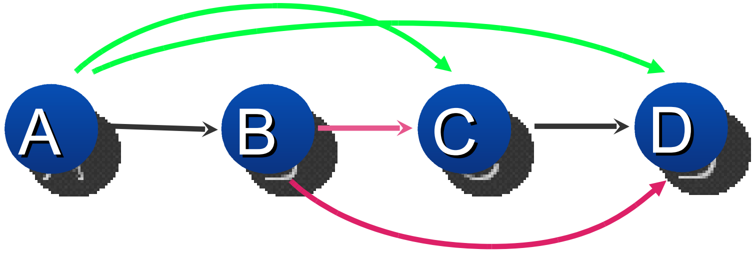
Split Half Decision

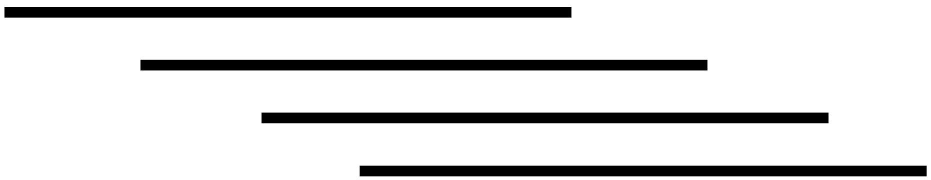


				Velvet
Max	1,205	1,492	2,083	2,423
N50	< 100	< 100	243	65
Cnt ≥ 1000	2	23	1,698	469
Sum ≥ 1000	2,258	25,348	1,959,690	546,205
Cnt ≥ 100	479,249	560,161	1,105,705	465,886
Sum ≥ 100	75,486,417	98,847,046	237,546,208	102,678,662

String graph assembly

- Similar problems with deBruijn
- Some new challenges:
  - – transitive edge removal
  - – can be handled through parallel set operations:





Graph
A->B,C,D
B->C,D
C->D

Map
A->B,C,D => (B; A->B,C,D) (A; A->B,C,D)
B->C,D => (B; B->C,D) (C; B->C,D)

Reduce
(B; A->B,C,D) (B; B->C,D) => A->B

Conclusions

- Trading CPUs for RAM works: data-intensive computing is possible in the cloud

- Embarassingly parallel problems – fairly easy (though not trivial)

- Load balancing tricky (esp. in assembly)

- Network bandwidth is critical

BUT

Biologists ecstatic!

Acknowledgments

Ben Langmead (now at JHU)

Cole Trapnell

Dan Sommer

Steven Salzberg

Jimmy Lin

Fritz McCall

Miron Livny (U. Wisconsin)

Deepak Singh (Amazon)

NSF IIS-0844494