Rapid Parallel Genome Indexing using MapReduce

Rohith Menon, Goutham Bhat & Michael Schatz*

June 8, 2011
HPDC'11/MapReduce

Outline

1. Brief Overview of DNA Sequencing

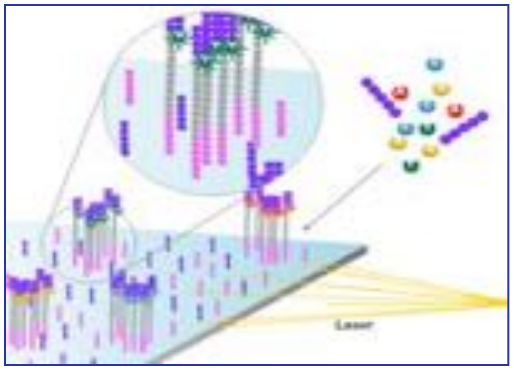
2. Genome Indexing
   - Serial, Basic MR, Optimized MR

3. Experimental Results

Molecular Biology & DNA Sequencing

Genome of an organism encodes the genetic information in long sequence of 4 DNA nucleotides: ACGT

- Bacteria: ~3 million bp
- Humans: ~3 billion bp



Current DNA sequencing machines can sequence billions of short (25-500bp) reads from random positions

- Per-base error rate estimated at 1-2% (Simpson *et al*, 2009
- Requires smart systems to analyze the sequences

ATCTGATAAGTCCCAGGACTTCAGT

GCAAGGCAAACCCGAGCCCAGTTT

TCCAGTTCTAGAGTTTCACATGATC

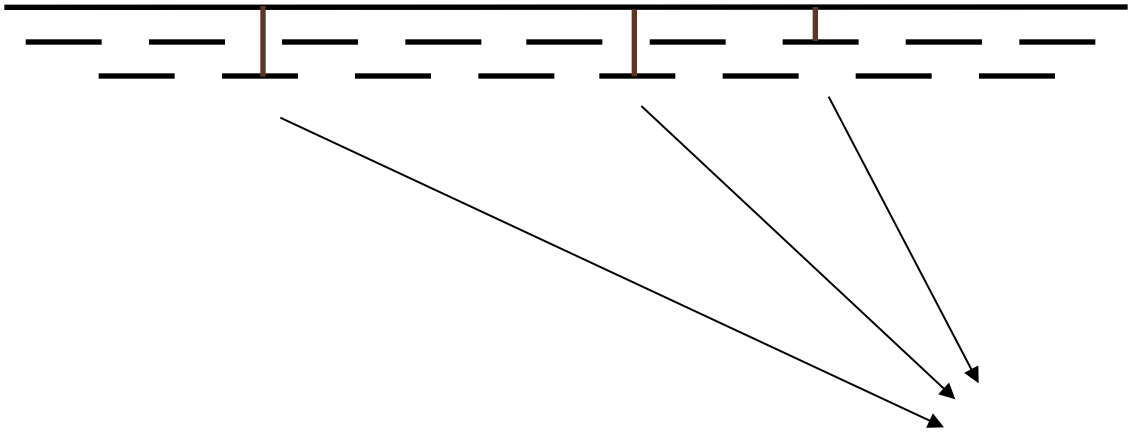
GGAGTTAGTAAAAGTCCACATTGAG

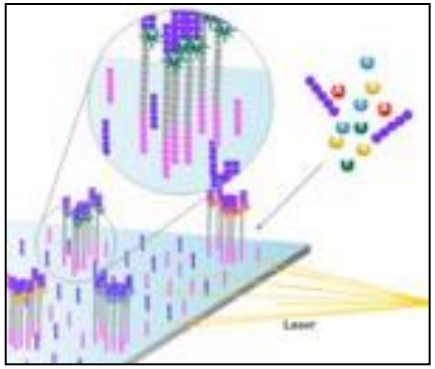
Modern Biology requires Computational Biology

- Individual reads have very little information
- World-wide sequencing capacity exceeds 12Pbp/year

Personal Genomics

How does your genome compare to Craig's?





Heart Disease

Cancer

Cool under fire

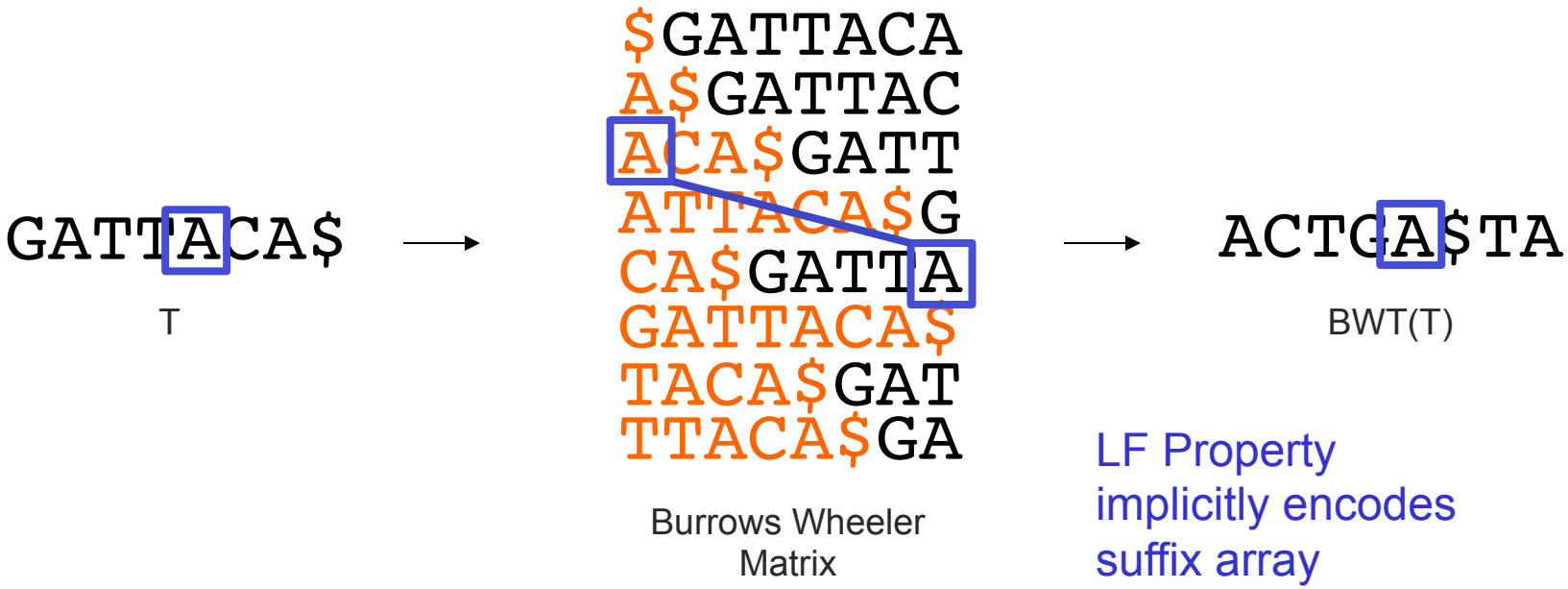
Accelerating Short Read Mapping

- Naïve Read Mapping is hopelessly slow
  - 1 billion 100bp reads x 3 billion positions
- Use an ***index*** to accelerate the search
  - Skip to "S" to lookup Schatz in the phonebook
  - No word boundaries in the genome, so consider every possible word/suffix
- The ***Suffix Array*** (Manber & Myers, 1991) is one of the most popular index structures
  - Lexicographically sorted list of suffixes
  - Fast binary search lookups: O(lg n) = 32 probes / read
  - Relatively space efficient: O(n lg n) =15GB / genome
  - Core index for Vmatch (http://www.vmatch.de/) and many other applications

Suffix array of "GATTACA$"

7	$
6	A$
4	ACA$
1	ATTACA$
5	CA$
0	GATTACA$
3	TACA$
2	TTACA$

Burrows-Wheeler Transform



- Suffix Array is tight, but much larger than genome
  - BWT is a reversible permutation of the genome based on the suffix array
  - Fast search and linear space requirements
  - Core index for Bowtie (Langmead *et al.*, 2009) and most recent short read mapping applications

A block sorting lossless data compression algorithm.
Burrows M, Wheeler DJ (1994) *Digital Equipment Corporation, Palo Alto, CA* 1994, Technical Report 124

Index Construction

- Naïve Suffix Array Construction
  - $O(n^2 \lg n) = O(n \lg n)$ comparisons x $O(n)$ per comparison

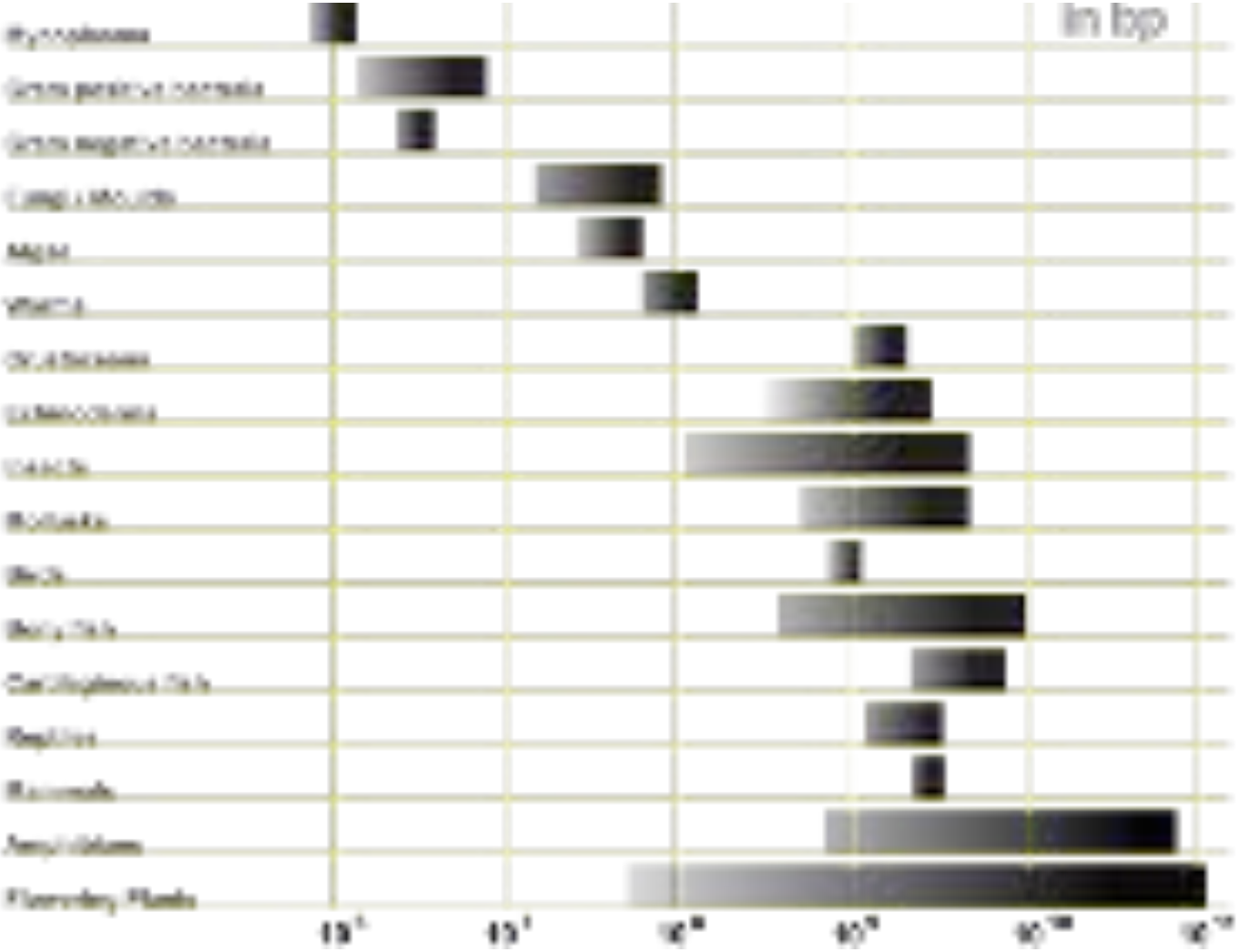
- Linear time Suffix Array Construction
  - Original: Construct ***suffix tree*** -> traverse tree (Weiner, 1973)
  - Recent: ***Difference Cover / DC3*** (Karkkainen *et al.*, 2006)
    - Intuition
      - O(1) to order suffixes a & b if we know order of a+1 & b+1
      - Recursively order 2/3G to order remaining 1/3 in O(1)

- BWT trivially constructed from SA or from (slower) counting techniques

- The leading methods require ***several hours*** for each mammalian genome
  - Parallel methods not generally applied because of the requirement for very fast interconnect (Kulla *et al.*, 2007)

Indexing Challenges

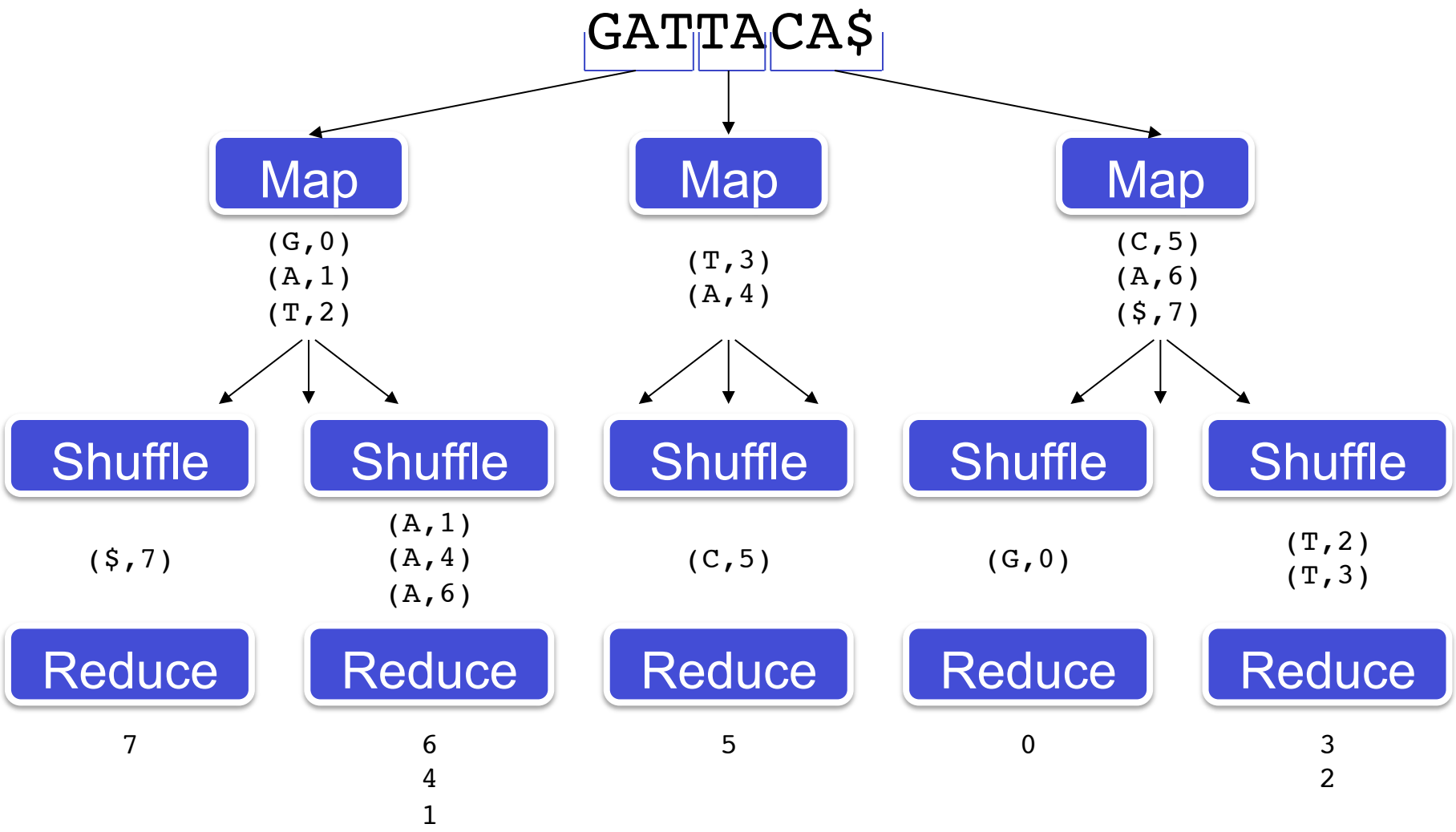
Sequencing underway for great numbers of very large genomes



http://en.wikipedia.org/wiki/File:Genome_Sizes.png

Basic Construction with MapReduce

Partition suffixes in lexicographically distinct bins, independently sort each bin



Optimizations

Hadoop Optimizations

1. Shuffle "bare" indices to reduce shuffle volume
2. Use Sampling Partitioner to optimize load balance
   - Inspired by TotalOrderPartitioner from SortBenchmark.org
3. Run length encode bin boundaries to reduce size
   - AAA….AAAG => A:10000|G:1

See paper for gory details

Reducer Optimizations

1. Recursive Bucket Sort using first p characters (p=15)
2. Precompute single nucleotide repeat length
   - Linear time sort of long simple repeats AAA….AAAG
   - Accelerate comparing simple repeats AAA…AAAGAAA…AAAC
3. Rank memoization (inspired by DC3 algorithm)
   - Use relative rank of suffixes a,b to accelerate comparison of a-d,b-d

Experimental Evaluation

http://code.google.com/p/genome-indexing

- Implementation
  - Java and JNI/C++
  - Genome in shared memory

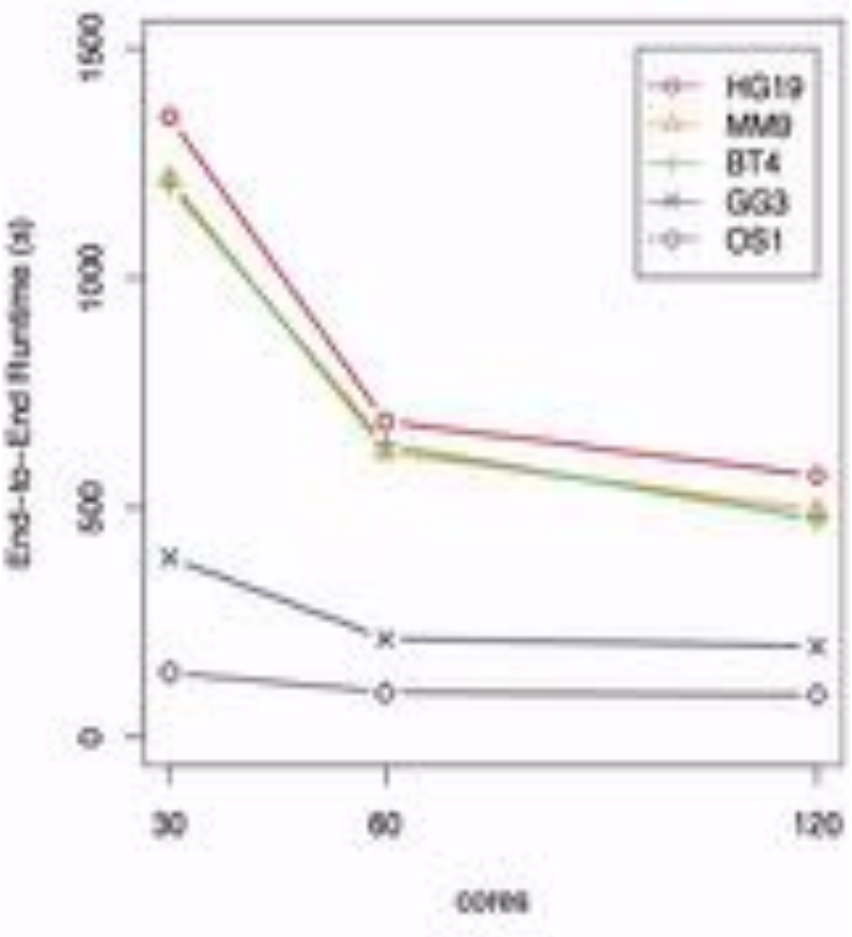
- Testbed: Amazon EC2
  - High-Memory double extra large instances ($1 / hour).
  - 4 HT cores @ 3.2 EC2 compute units
  - 34.2G RAM, 850G Disk
  - Hadoop 0.20.2, VM Image: AMI-6AA34003
  - Max cluster size: 21 (1 master and 20 drones)

Genomes evaluated

Name	Genome	Build	Length (nt)
HG19	Human (*Homo sapiens*)	19	3,095,677,412
MM9	Mouse (*Mus musculus*)	9	2,654,895,218
BT4	Cow (*Bos taurus*)	4	2,634,413,324
GG3	Chicken (*Gallus gallus*)	3	1,031,883,471
OS1	Rice (*Oryza sativa*)	1	370,792,118

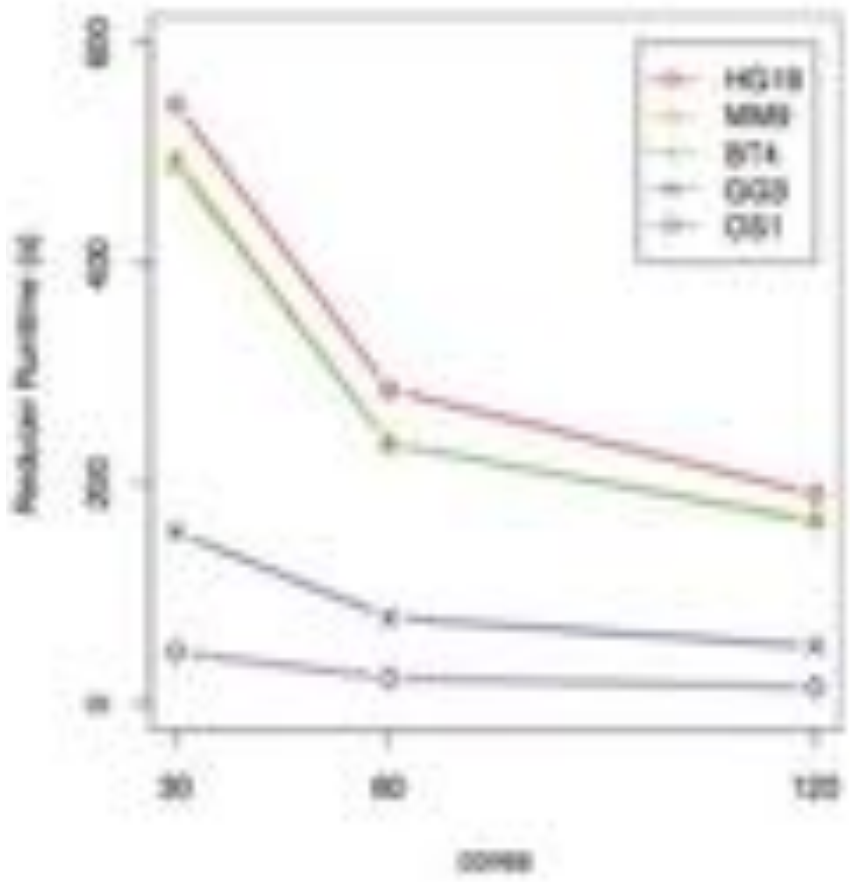
End-to-End Performance

- Evaluate performance using increasing numbers of cores
  - 15x speedup over Bowtie
  - 9x speedup over Vmatch
- Performance beyond 60 cores is limited by Hadoop overhead.
  - 120C cluster requires 398s to scan human genome using HashPartitioner and IdentityReducer to write unsorted list of suffixes



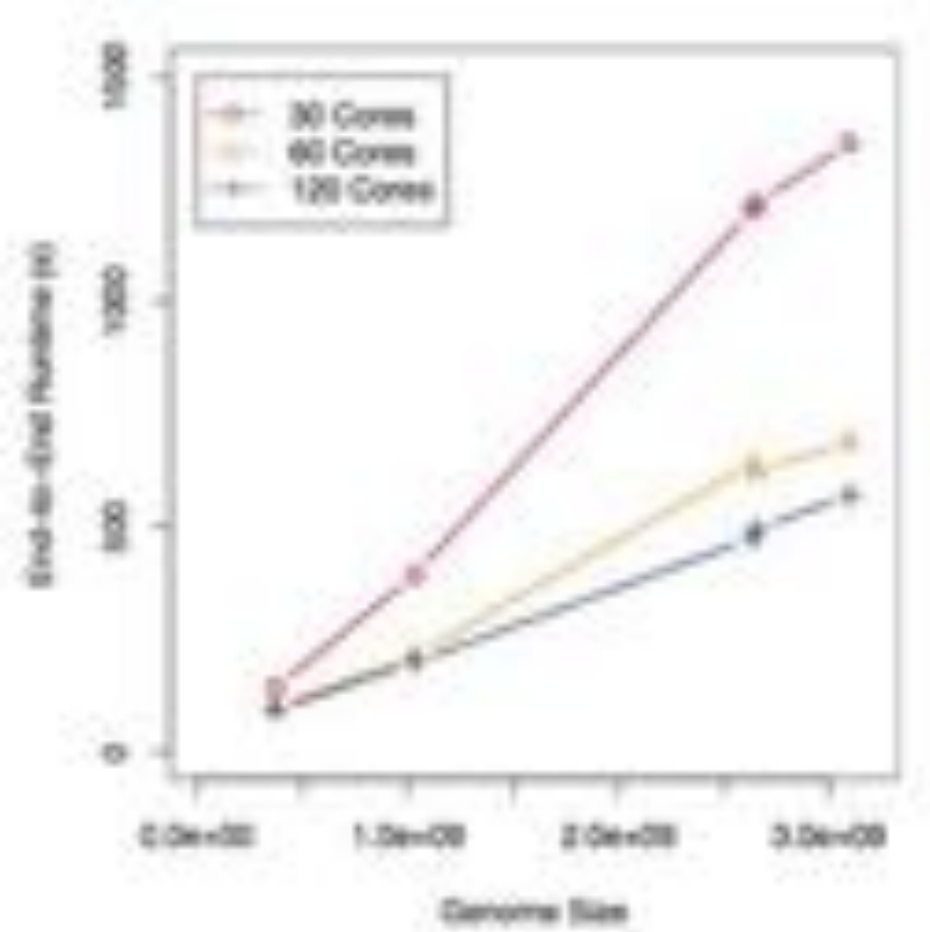
Reducer run time

- End-to-end runtime has substantial Hadoop overhead
- Meaure runtime of the reducer alone
  - Start: Index collection by reducer
  - End: SA written to local disk
- Reducer runtime improved performance through 120C



Genome Scaling Performance

- Evaluate performance using a fixed number of cores across the 5 genomes
- End-to-end runtime is ~linear with the size of the genome
  - Our performance optimizations are very effective on real genomes
  - 3Gbp Human genome takes ~9 minutes
  - Scaling to loblolly pine (24Gbp) should only take ~1hr 10 min



Hadoop for NGS Analysis

CloudBurst

Highly Sensitive Short Read Mapping with MapReduce

100x speedup mapping on 96 cores @ Amazon

http://cloudburst-bio.sf.net

(Schatz, 2009)

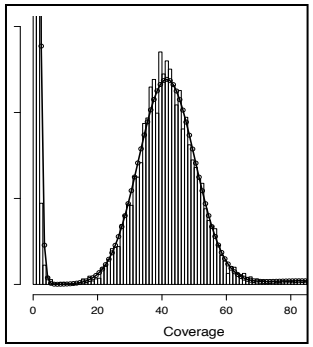
Crossbow

Searching for SNPs with Cloud Computing

Identify 3M SNPs in an afternoon

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

http://bowtie-bio.sf.net/crossbow/



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)

Contrail

Assembly of Large Genomes Using Cloud Computing

Quickly assemble the human genome with hundreds of commodity cores

(Schatz et al. 2011*)

http://contrail-bio.sf.net/

Summary

- Staying afloat in the data deluge means computing in parallel
  - Hadoop + Cloud computing is an attractive platform for large scale sequence analysis and computation
- Our algorithm has substantially accelerated a critical problem in computational biology
  - Conceptually straightforward, but required careful algorithm analysis and engineering
  - Current performance limited by Hadoop
- Future Work
  - Integration with Bowtie, BWA
  - Phased algorithm for low memory clusters, read indexing
  - Continue development of MapReduce-enabled algorithms for biology

Acknowledgements

Rohith Menon

Goutham Bhat

Ben Langmead

Steve Skiena

Adam Phillippy

amazon web services™

Thank You!

http://schatzlab.cshl.edu
@mike_schatz