High Performance Computing for DNA Sequence Alignment and Assembly

Michael C. Schatz

May 18, 2010
Stone Ridge Technology

Outline

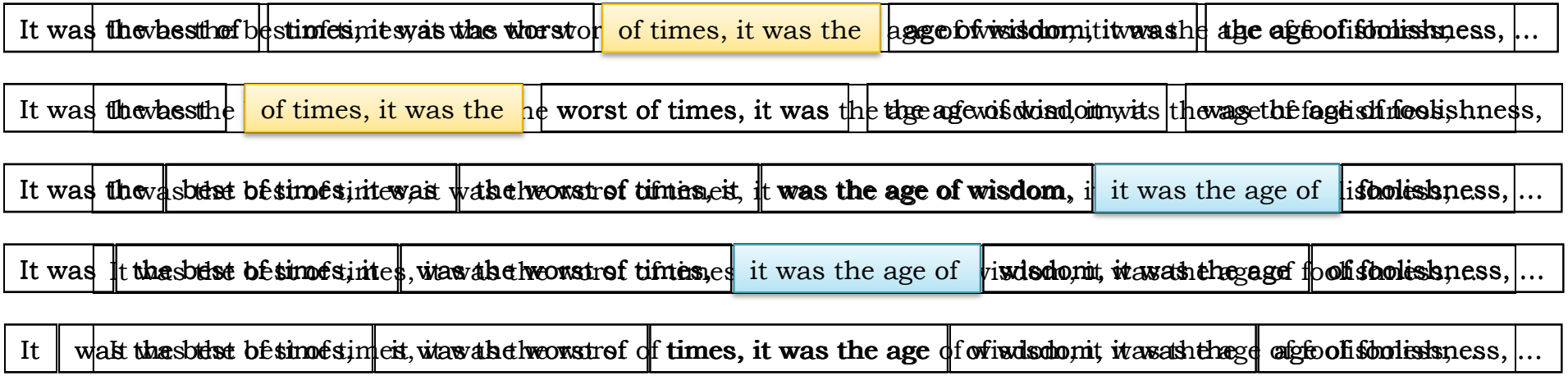
1. Sequence Analysis by Analogy

2. DNA Sequencing and Genomics

3. High Performance Sequence Analysis
    1. Read Mapping
    2. Mapping & Genotyping
    3. Genome Assembly

Shredded Book Reconstruction

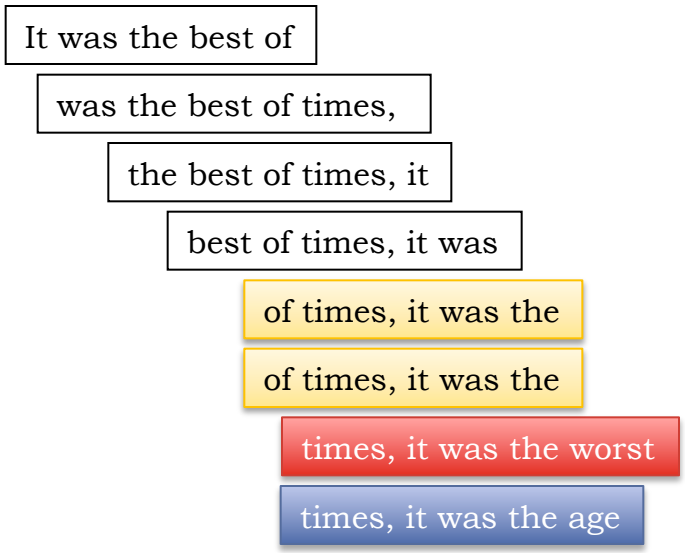
- Dickens accidentally shreds the first printing of A Tale of Two Cities
  - Text printed on 5 long spools



- 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



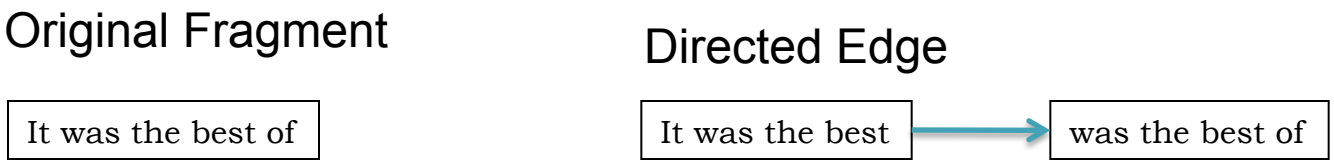
The repeated sequence make the correct reconstruction ambiguous

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

Model sequence reconstruction as a graph problem.

de Bruijn Graph Construction

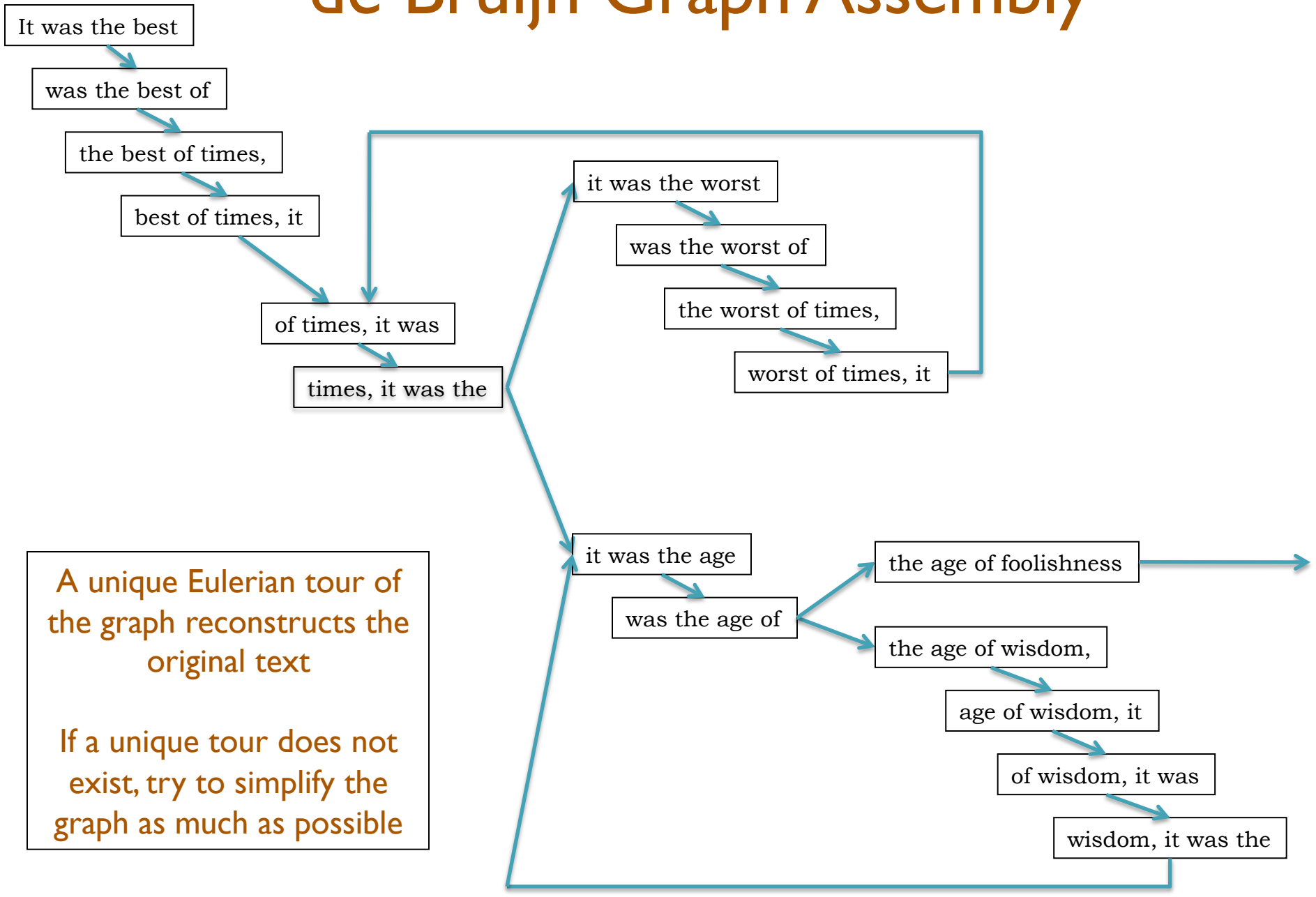
- $D_k = (V,E)$
  - V = All length-k subfragments (k < l)
  - E = Directed edges between consecutive subfragments
    - Nodes overlap by k-1 words



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

de Bruijn, 1946
Idury and Waterman, 1995
Pevzner, Tang, Waterman, 2001

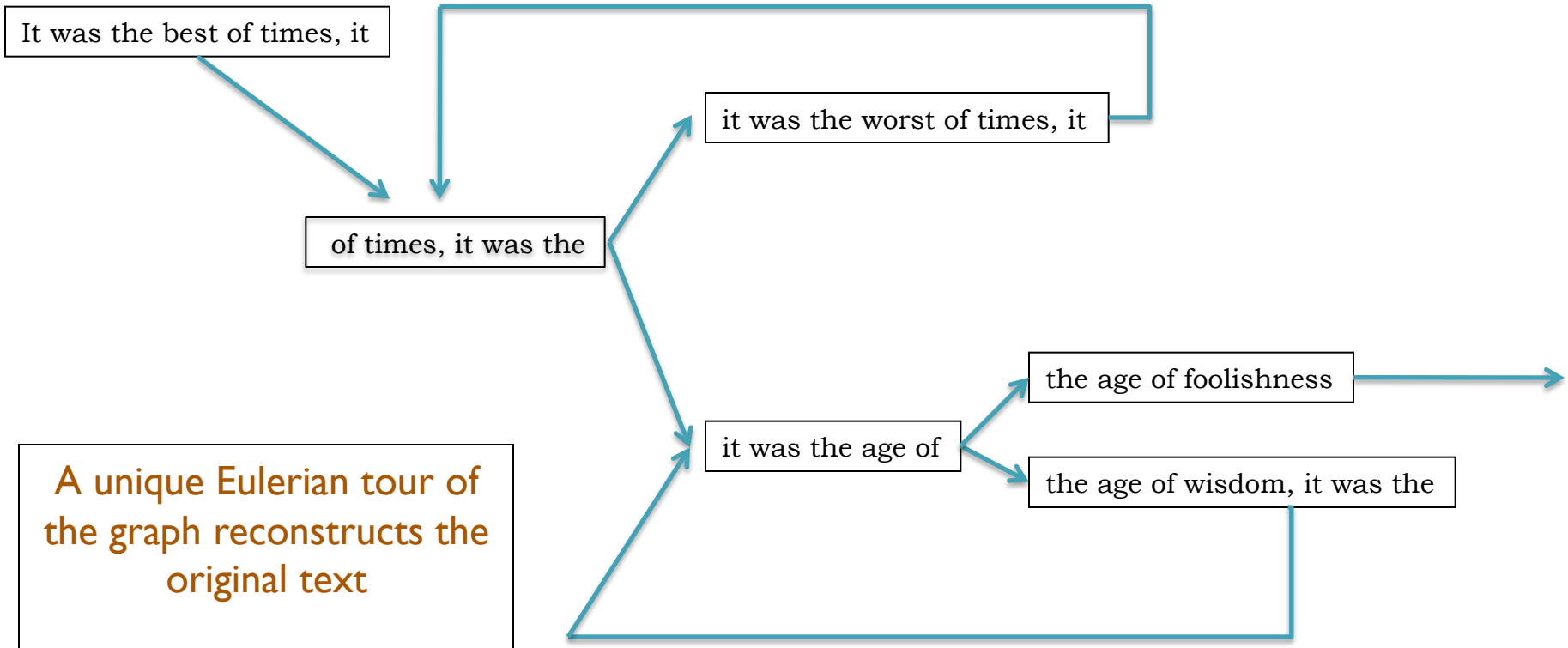
de Bruijn Graph Assembly



A unique Eulerian tour of the graph reconstructs the original text

If a unique tour does not exist, try to simplify the graph as much as possible

de Bruijn Graph Assembly

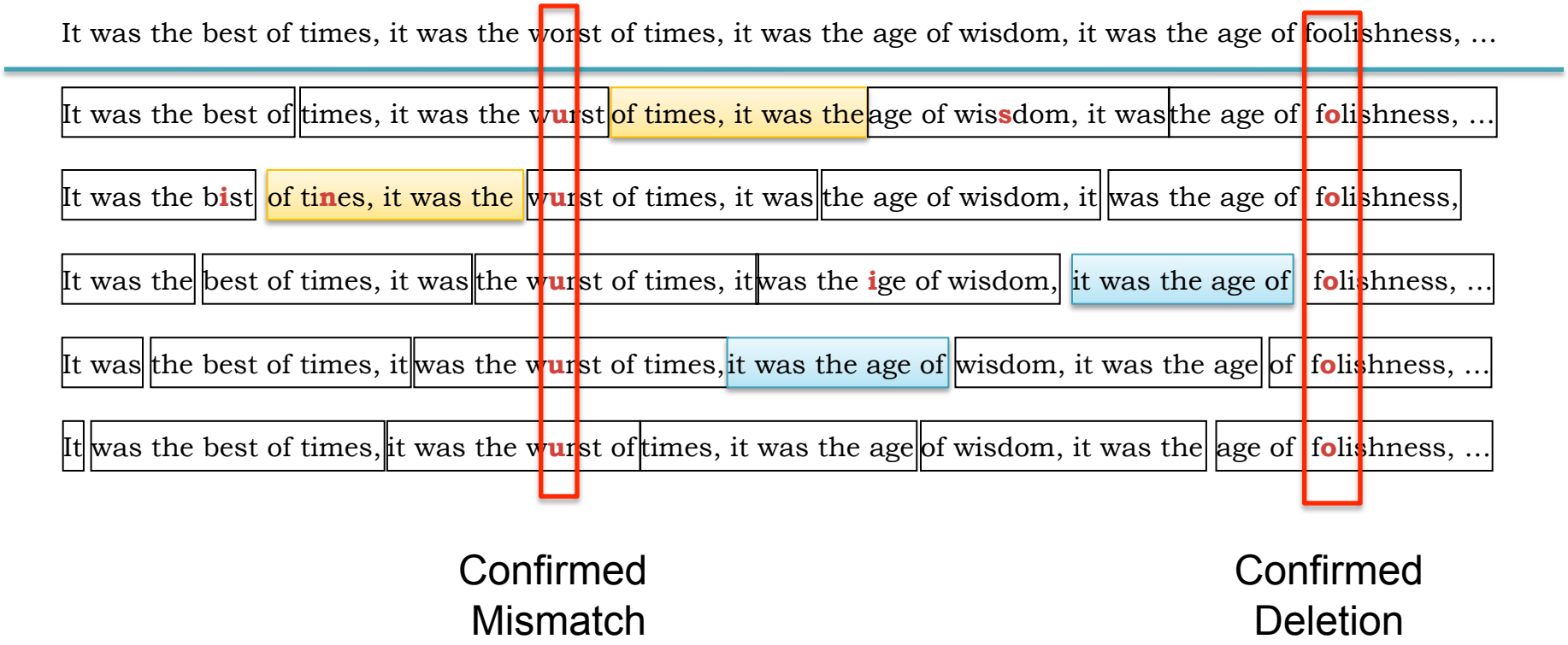


A unique Eulerian tour of the graph reconstructs the original text

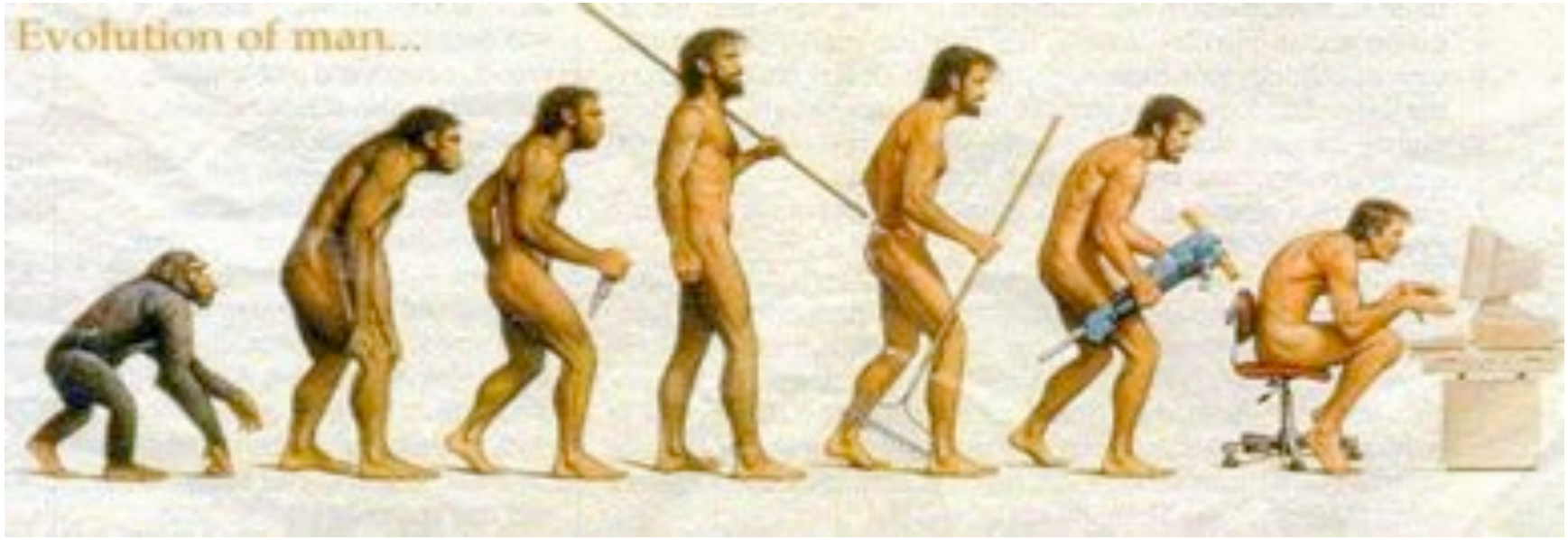
If a unique tour does not exist, try to simplify the graph as much as possible

Shredded Book Mapping

- Dickens searches for misprints in the shredded copies
  - Find the best match for each fragment
  - Has to account for random and systematic variations



Genomics and Evolution

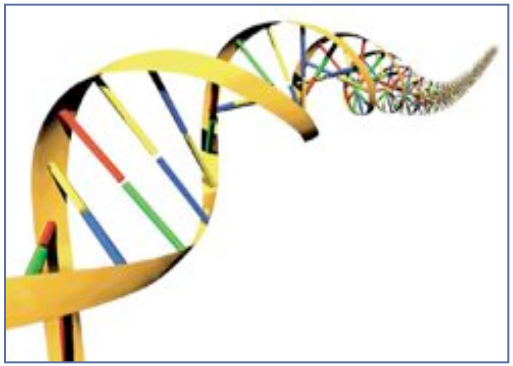


Your genome influences (almost) all aspects of your life

- Anatomy & Physiology: 10 fingers & 10 toes, organs, neurons
- Diseases: Sickle Cell Anemia, Down Syndrome, Cancer
- Psychological: Intelligence, Personality, Bad Driving
- Genome as a recipe, not a blueprint

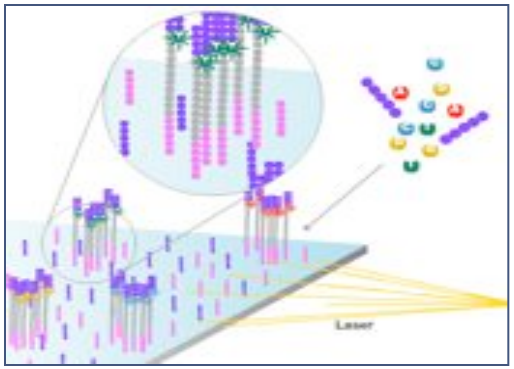
Like Dickens, we can only sequence small fragments of the genome

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 generate 1-2 Gbp of sequence per day, in millions of short reads

- Per-base error rate estimated at 1-2% (Simpson *et al*, 2009)
- Sequences originate from random positions of the genome
- Base calling transforms raw images into DNA sequences

ATCTGATAAGTCCCAGGACTTCAGT

GCAAGGCAAACCCGAGCCCAGTTT

TCCAGTTCTAGAGTTTCACATGATC

GGAGTTAGTAAAAGTCCACATTGAG

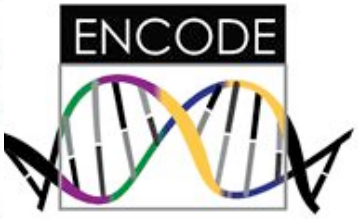
Recent studies of entire human genomes analyzed 3.3B (Wang, et al., 2008) & 4.0B (Bentley, et al., 2008) 36bp reads

- ~100 GB of compressed sequence data

The Evolution of DNA Sequencing

Year	Genome	Technology	Cost
2001	Venter *et al.*	Sanger (ABI)	$300,000,000
2007	Levy *et al.*	Sanger (ABI)	$10,000,000
2008	Wheeler *et al.*	Roche (454)	$2,000,000
2008	Ley *et al.*	Illumina	$1,000,000
2008	Bentley *et al.*	Illumina	$250,000
2009	Pushkarev *et al.*	Helicos	$48,000
2009	Drmanac *et al.*	Complete Genomics	$4,400

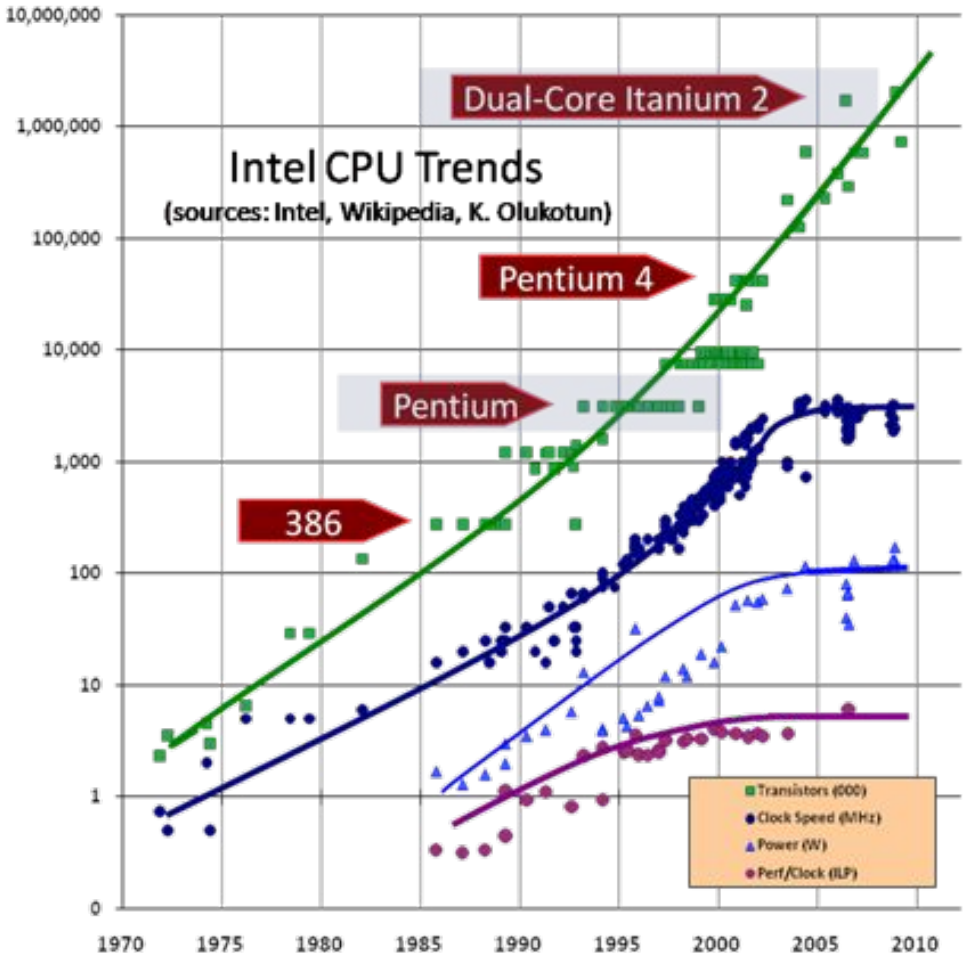
(Pushkarev *et al.*, 2009)



Critical Computational Challenges: Alignment and Assembly of Huge Datasets

Why HPC?

- Moore's Law is valid in 2010
  - But CPU speed is flat
  - Vendors adopting parallel solutions instead
- Parallel Environments
  - Many cores, including GPUs
  - Many computers
  - Many disks
- Why parallel
  - Need results faster
  - Doesn't fit on one machine



The Free Lunch Is Over: A Fundamental Turn Toward Concurrency in Software
Herb Sutter, http://www.gotw.ca/publications/concurrency-ddj.htm

Hadoop MapReduce

- 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

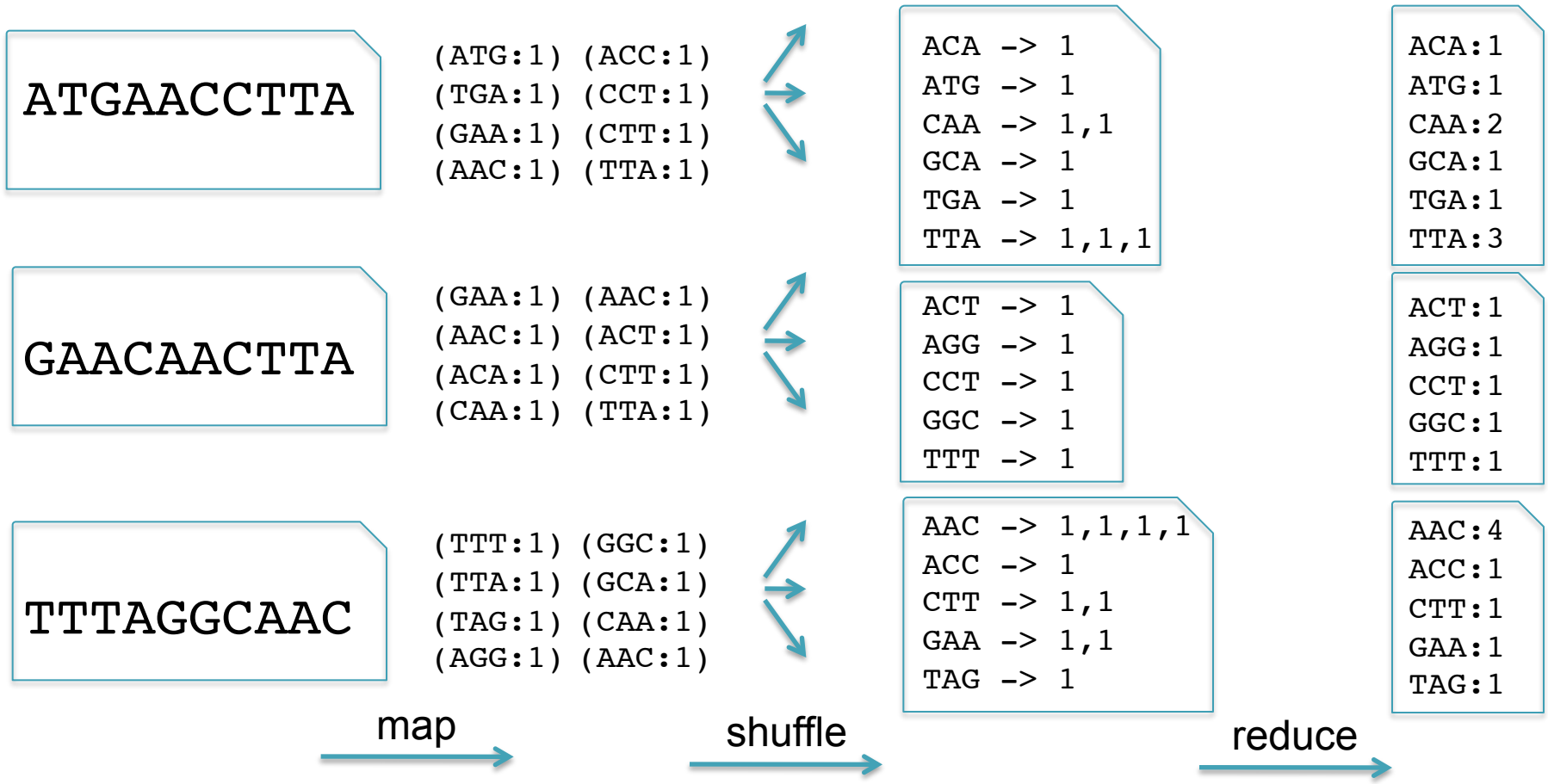
- Benefits
  - Scalable, Efficient, Reliable
  - Easy to Program
  - Runs on commodity computers

- Challenges
  - Redesigning / Retooling applications
    - Not Condor, Not MPI
    - Everything in MapReduce

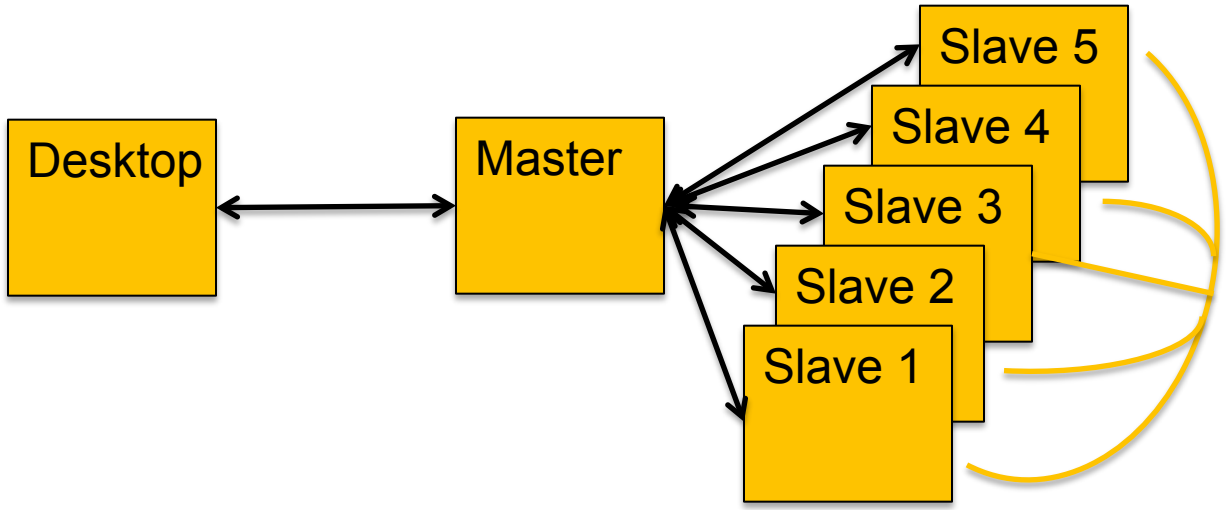
K-mer Counting

- 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

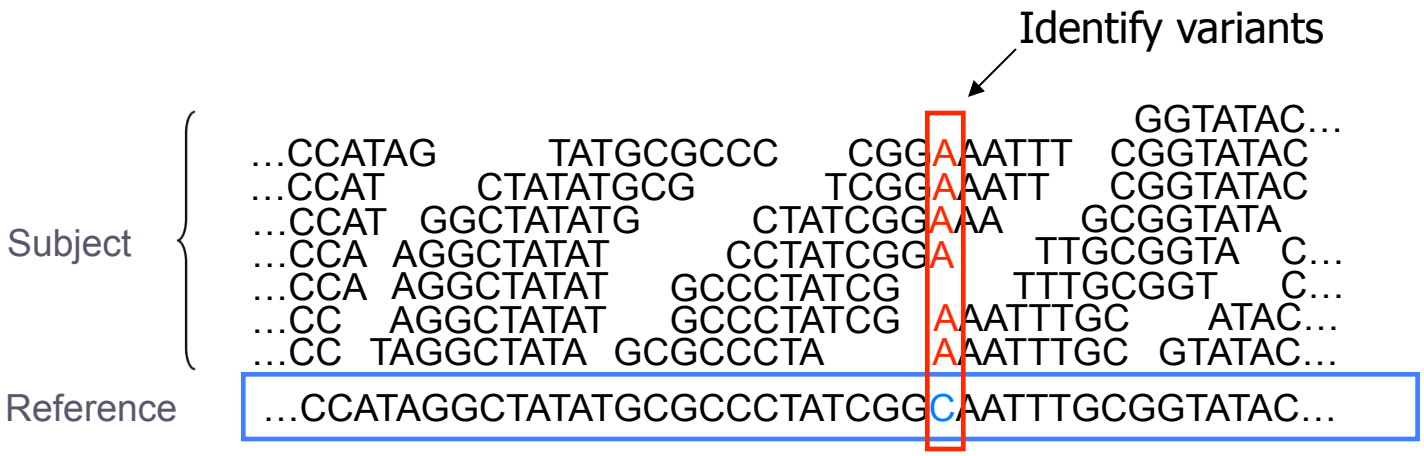


Hadoop Architecture



- Hadoop Distributed File System (HDFS)
  - Data files partitioned into large chunks (64MB), replicated on multiple nodes
  - NameNode stores metadata information (block locations, directory structure)
- Master node (JobTracker) schedules and monitors work on slaves
  - Computation moves to the data, rack-aware scheduling
- Hadoop MapReduce system won the 2009 GreySort Challenge
  - Sorted 100 TB in 173 min (578 GB/min) using 3452 nodes and 4x3452 disks

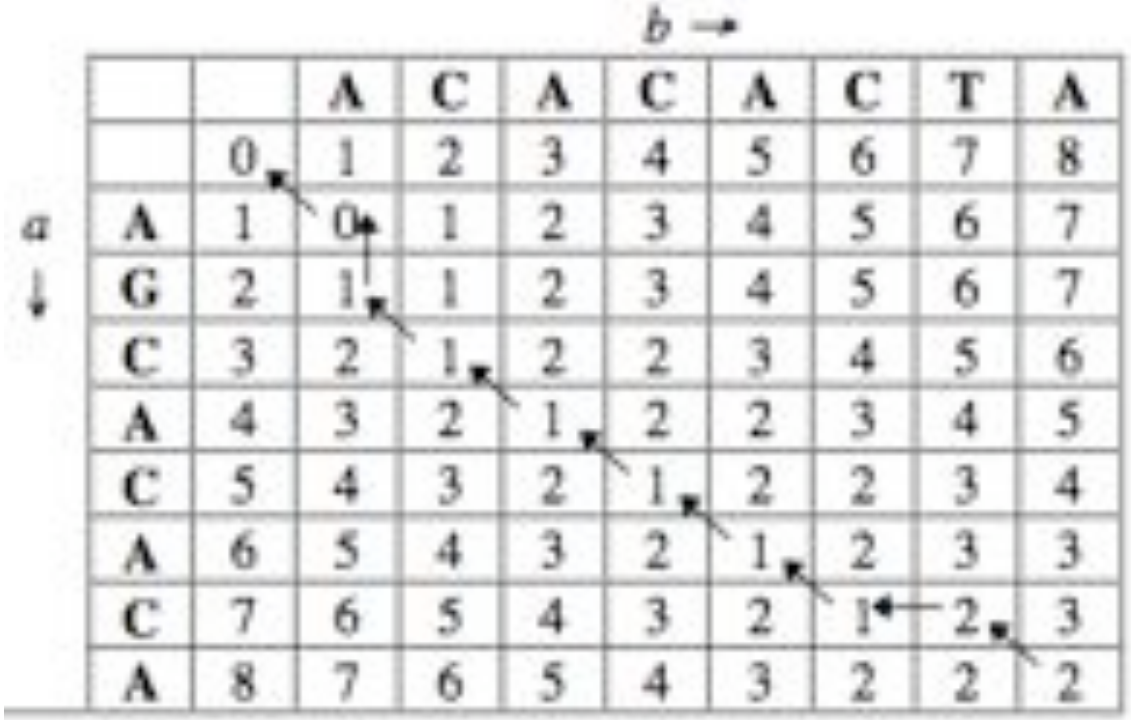
Short Read Mapping



- Given a reference and many subject reads, report one or more "good" end-to-end alignments per alignable read
  - Find where the read most likely originated
  - Fundamental computation for many assays
    - Genotyping
    - Structural Variations
    - RNA-Seq
    - Chip-Seq
    - Methyl-Seq
    - Hi-C-Seq

- Desperate need for scalable solutions
  - Single human requires >1,000 CPU hours / genome

Sequence Alignment with Dynamic Programming



			A	C	A	C	A	C	T	A
		0	1	2	3	4	5	6	7	8
a ↓	A	1	0	1	2	3	4	5	6	7
	G	2	1	1	2	3	4	5	6	7
	C	3	2	1	2	2	3	4	5	6
	A	4	3	2	1	2	2	3	4	5
	C	5	4	3	2	1	2	2	3	4
	A	6	5	4	3	2	1	2	3	3
	C	7	6	5	4	3	2	1	2	3
	A	8	7	6	5	4	3	2	2	2

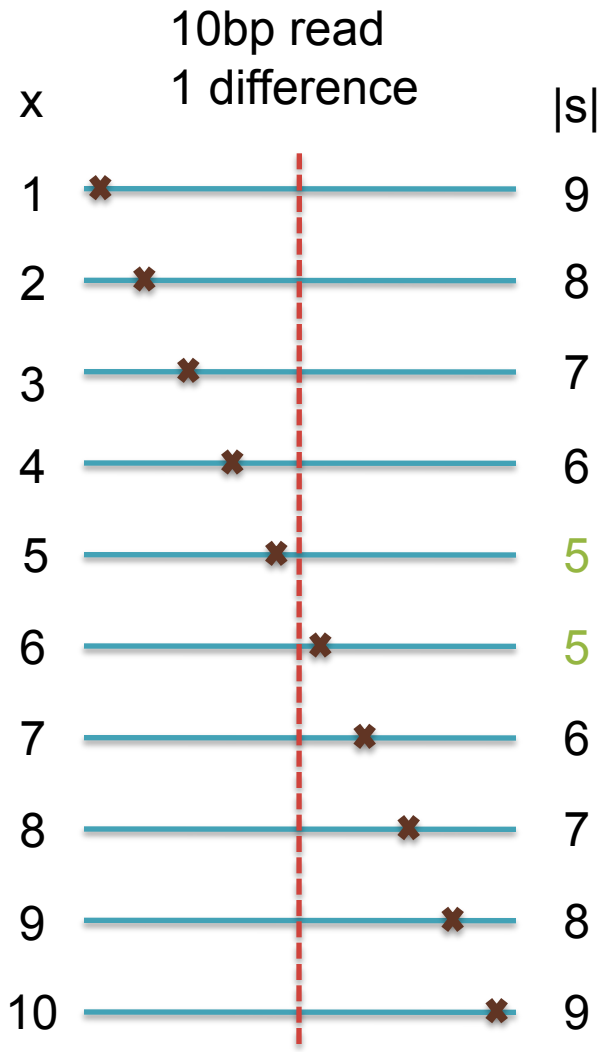
b →

```
A-CACACTA
AGCACAC-A
```

$$D(i,j) = \min \{ \; D(i-1,j) + 1, \quad D(i,j-1) + 1, \quad D(i-1,j-1) + \delta(S(i),T(j)) \; \}$$

Seed and Extend

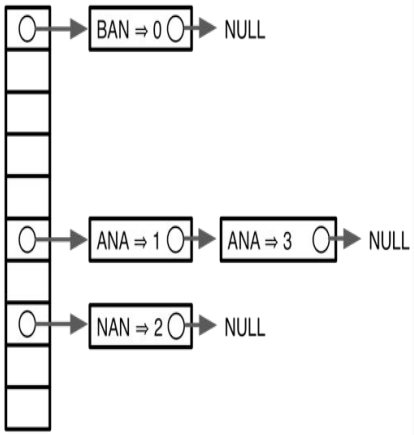
- Highly similar alignments must have significant exact seeds
  - Use exact alignments to seed search for longer in-exact alignments
  - Pigeon hole principle: if a read matches someplace with k differences, one of its k+1 chunks must match exactly
- BLAST (Altschul et al., 1990)
  - Catalog fixed length substrings (k-mers) as seeds
  - Use Smith-Waterman dynamic programming algorithm to extend seeds into longer in-exact alignments
  - Arguably the most widely used tool in computational biology
    - 10s of thousands of citations



Indexing

- Genomes are too large for dynamic programming
  - Use an index to find candidate seeds to extend

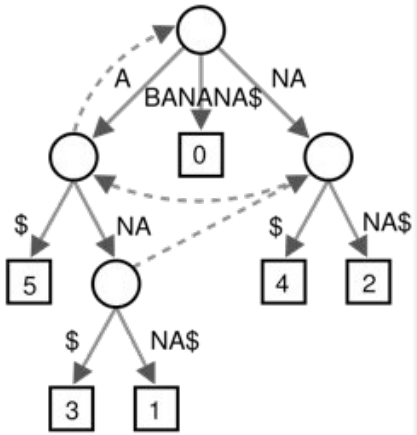
Hash Table (>15 GB)



BLAST, MAQ, ZOOM, RMAP, CloudBurst

Fixed length, irregular access

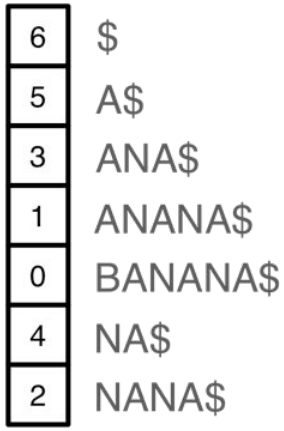
Suffix Tree (>51 GB)



MUMmer, MUMmerGPU

Variable length, Pointer Jumping

Suffix Array (>15 GB)



Vmatch, PacBio Aligner

Variable length, Binary Search

Burrows-Wheeler (3 GB)

$BANANA
A$BANAN
ANA$BAN
ANANA$B
BANANA$
NA$BANA
NANA$BA

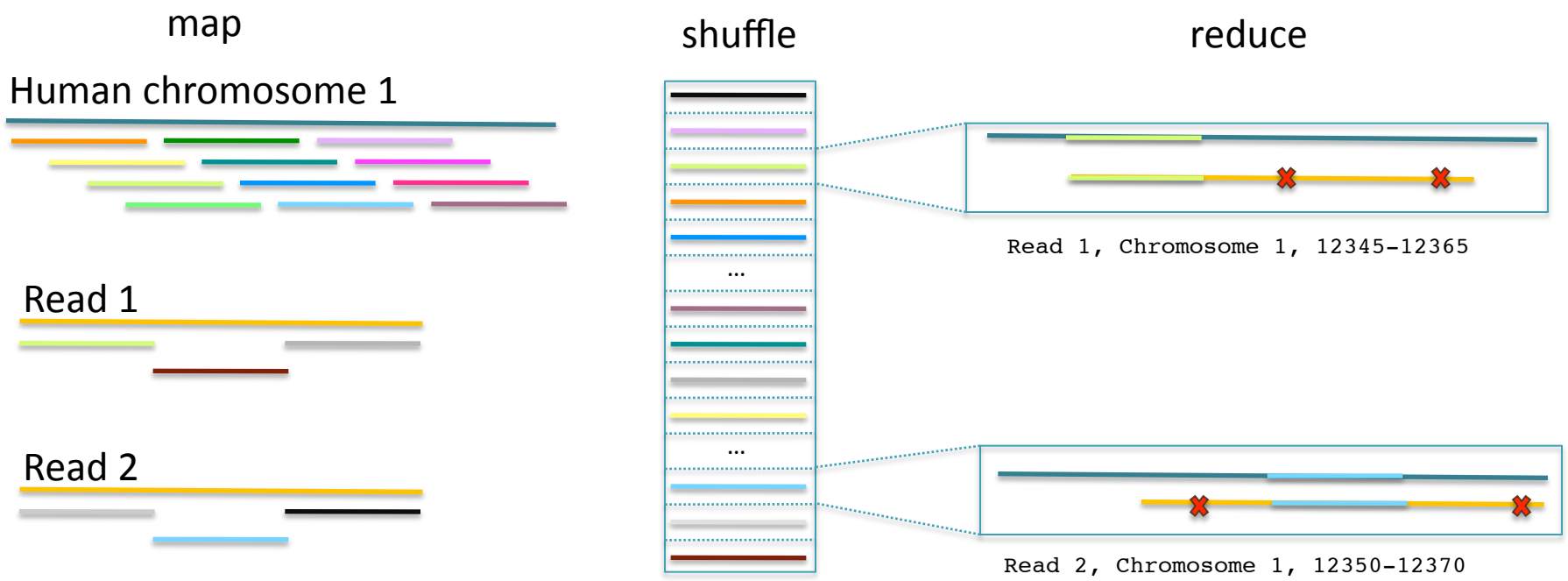
Bowtie, BWA

Variable length, Range Queries

CloudBurst

http://cloudburst-bio.sourceforge.net

- Leverage Hadoop to build a distributed inverted index of k-mers and find end-to-end alignments
- 100x speedup over RMAP with 96 cores at Amazon EC2



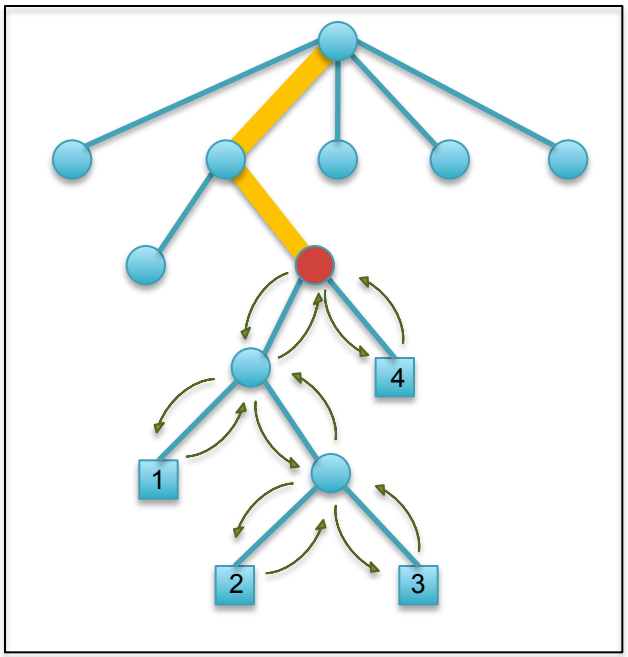
CloudBurst: Highly Sensitive Read Mapping with MapReduce.
Schatz MC (2009) *Bioinformatics*. 25:1363-1369

MUMmerGPU

http://mummergpu.sourceforge.net

- Map many reads simultaneously on a GPU
  - Index reference using a suffix tree
  - Find matches by walking the tree
  - Find coordinates with depth first search

- Performance on nVidia GTX 8800
  - Match kernel was ~10x faster than CPU
  - Print kernel was ~4x faster than CPU
  - End-to-end runtime ~4x faster than CPU

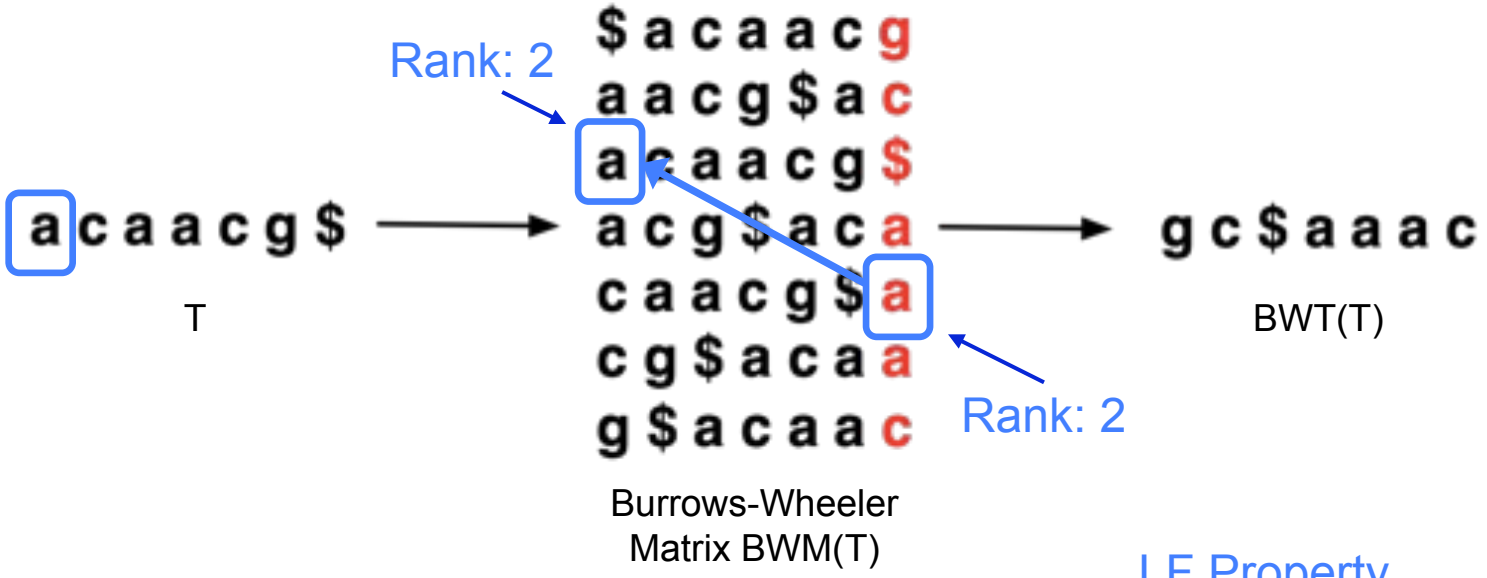


High-throughput sequence alignment using Graphics Processing Units.
Schatz, MC*, Trapnell, C*, Delcher, AL, Varshney, A. (2007) *BMC Bioinformatics* 8:474.

Optimizing data intensive GPGPU computations for DNA sequence alignment.
Trapnell C*, Schatz MC*. (2009) *Parallel Computing.* 35(8-9):429-440.

Burrows-Wheeler Transform

- Reversible permutation of the characters in a text



LF Property implicitly encodes Suffix Array

- BWT(T) is the index for T

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

Bowtie: Ultrafast Short Read Aligner

- Quality-aware backtracking of BWT to rapidly find the best alignment(s) for each read

- BWT precomputed once, easy to distribute, and analyze in RAM
  - 3 GB for whole human genome

- Support for paired-end alignment, quality guarantees, etc…
  - Langmead B, Trapnell C, Pop M, Salzberg SL. (2009) Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biology* 10:R25.

Bowtie algorithm

Reference

BWT(Reference)

Query:
A A T G A T A C G G C G A C C A C C G A G A T C T A

Bowtie algorithm

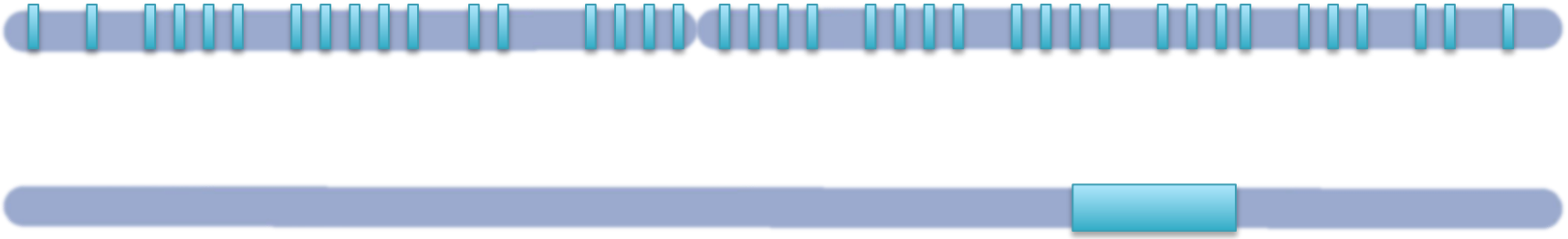
Reference

BWT(Reference)

Query:
A A T G A T A C G G C G A C C A C C G A G A T C T A

Bowtie algorithm

Reference

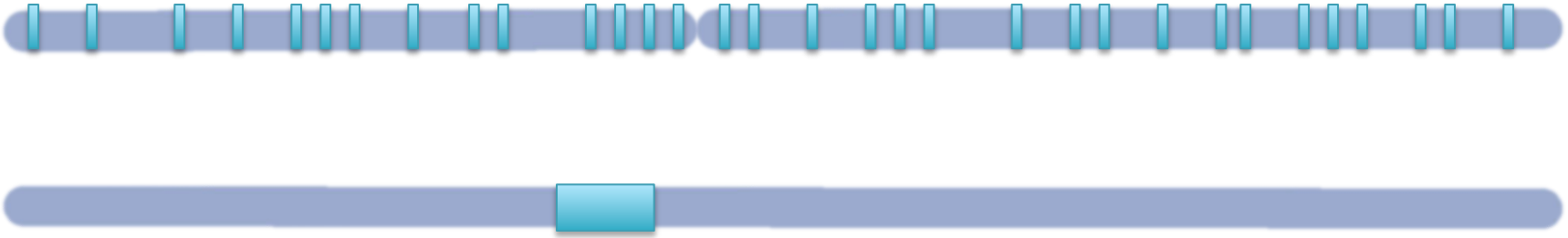


BWT(Reference)

Query:
A A T G A T A C G G C G A C C A C C G A G A T C T A

Bowtie algorithm

Reference

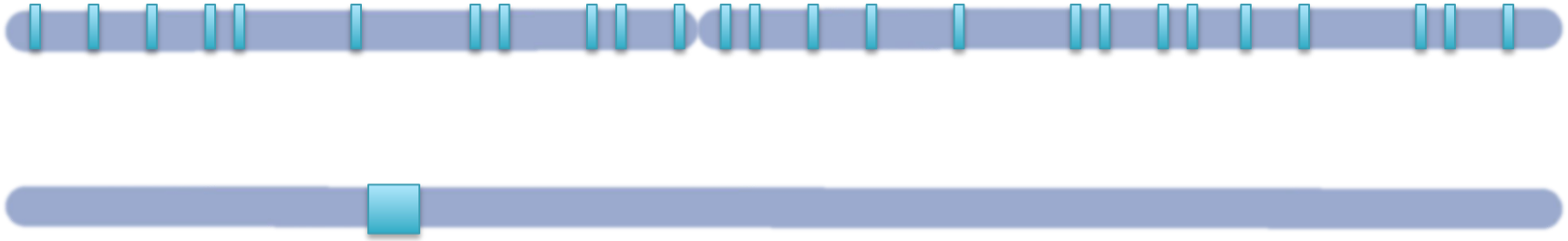


BWT(Reference)

Query:
A A T G A T A C G G C G A C C A C C G A G A T C T A

Bowtie algorithm

Reference



BWT(Reference)

Query:

A A T G A T A C G G C G A C C A C C G A G A T C T A

Bowtie algorithm

Reference

BWT(Reference)

Query:

A A T G A T A C G G C G A C C A C C G A G A T C T A

Bowtie algorithm

Reference

BWT(Reference)

Query:
AATG ATACGGCGACCACCGAGATCTA

Bowtie algorithm

Reference

BWT(Reference)

Query:

A A T G T T A C G G C G A C C A C C G A G A T C T A

Bowtie algorithm

Reference

BWT(Reference)

Query:

A A T G T T A C G G C G A C C A C C G A G A T C T A

Comparison to MAQ & SOAP

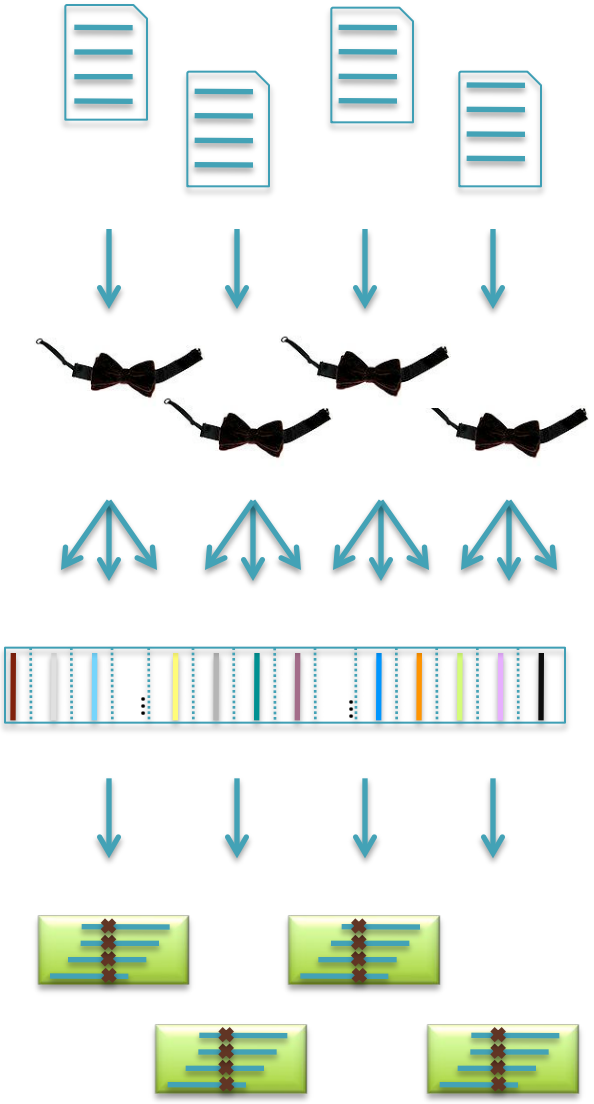
	CPU time	Wall clock time	Reads mapped per hour (millions)	Peak virtual memory footprint (MB)	Bowtie speedup	Reads aligned (%)
Bowtie –v 2 (server)	15m:07s	15m:41s	33.8	1,149	-	67.4
SOAP (server)	91h:57m:35s	91h:47m:46s	0.10	13,619	351x	67.3
Bowtie (server)	17m:58s	18m:26s	28.8	1,353	-	71.9
Bowtie --best (server)	46m:54s	47m:23s	11.2	2,383	2.6x	72.0
Maq (server)	32h:56m:53s	32h:58m:39s	0.27	804	107x	74.7

Performance and sensitivity of Bowtie v0.9.6, SOAP v1.10, Maq v0.6.6 when aligning 8.84M reads from the 1000 Genome project [NCBI Short Read Archive:SRR001115] trimmed to 35 base pairs. The "soap.contig" version of the SOAP binary was used. SOAP could not be run on the PC because SOAP's memory footprint exceeds the PC's physical memory. For the SOAP comparison, Bowtie was invoked with "-v 2" to mimic SOAP's default matching policy (which allows up to 2 mismatches in the alignment and disregards quality values). For the Maq comparison Bowtie is run with its default policy, which mimics Maq's default policy of allowing up to 2 mismatches in the first 28 bases and enforcing an overall limit of 70 on the sum of the quality values at all mismatched positions. To make Bowtie's memory footprint more comparable to Maq's, Bowtie is invoked with the "-z" option in all experiments to ensure only the forward or mirror index is resident in memory at one time.

Crossbow

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

- Align billions of reads and find SNPs
  - Reuse software components: Hadoop Streaming
- Map: Bowtie (Langmead *et al.*, 2009)
  - Find best alignment for each read
  - Emit (chromosome 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



Performance in Amazon EC2

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

	Asian Individual Genome		
Data Loading	3.3 B reads	106.5 GB	$10.65
Data Transfer	1h :15m	40 cores	$3.40
Setup	0h : 15m	320 cores	$13.94
Alignment	1h : 30m	320 cores	$41.82
Variant Calling	1h : 00m	320 cores	$27.88
End-to-end	4h : 00m		$97.69

Analyze an entire human genome for ~$100 in an afternoon.
Accuracy validated at >99%

Searching for SNPs with Cloud Computing.
Langmead B, Schatz MC, Lin J, Pop M, Salzberg SL (2009) *Genome Biology.*

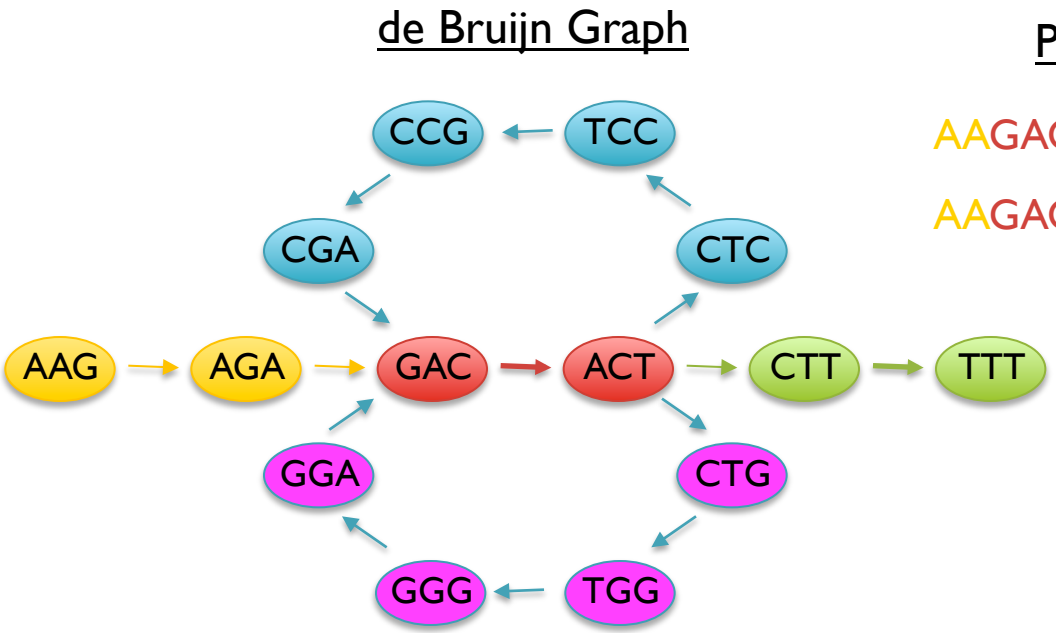
Hardware Accelerated Mapping

	Complexity	Index Size	Access Style
Dynamic Programming	Very Simple	N/A	Regular Grid
Seed Hash Table	Simple	Moderate	Random Access
Suffix Tree	Moderate	Large	Pointer Jumping
Suffix Array	Moderate	Moderate	Binary Search
BWT	Difficult	Small	Range Queries

Short Read Assembly

Reads

AAGA
ACTT
ACTC
ACTG
AGAG
CCGA
CGAC
CTCC
CTGG
CTTT
…



Potential Genomes

AAGACTCCGACTGGGACTTT

AAGACTGGGACTCCGACTTT

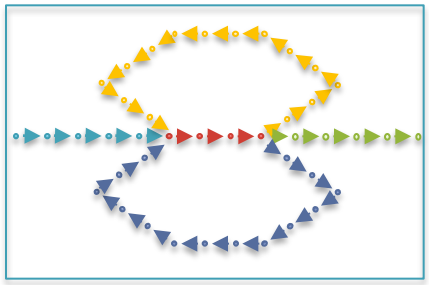
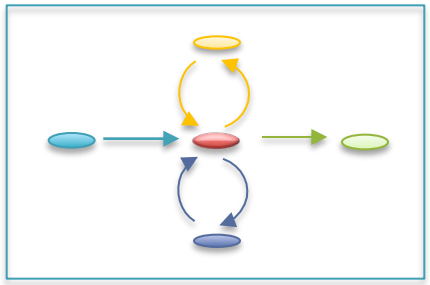
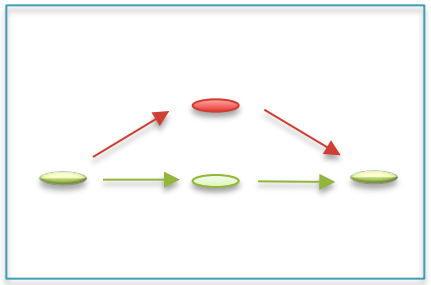
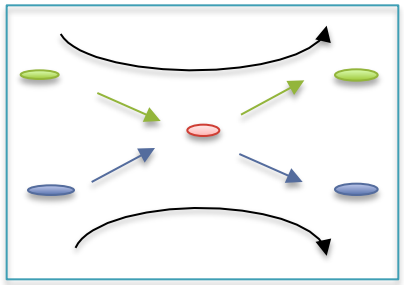
- Genome assembly as finding an Eulerian tour of the de Bruijn graph
  - Human genome: >3B nodes, >10B edges
- The new short read assemblers require tremendous computation
  - Velvet (Zerbino & Birney, 2008) serial: > 2TB of RAM
  - ABySS (Simpson *et al.*, 2009) MPI: 168 cores x ~96 hours
  - SOAPdenovo (Li *et al.*, 2010) pthreads: 40 cores x 40 hours, >140 GB RAM

Contrail

http://contrail-bio.sourceforge.net

De Novo Assembly of the Human Genome

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

	Initial	Compressed	Error Correction	Resolve Repeats
				
N	>7 B	>1 B	4.2 M	4.1 M
Max	27 bp	303 bp	20,594 bp	20,594 bp
N50	27 bp	<100 bp	995 bp	1050 bp

Assembly of Large Genomes with Cloud Computing.
Schatz MC, Sommer D, Kelley D, Pop M, *et al. In Preparation.*

Summary

"NextGen sequencing has completely outrun the ability of good bioinformatics people to keep up with the data and use it well… We need a MASSIVE effort in the development of tools for "normal" biologists to make better use of massive sequence databases."

Jonathan Eisen – JGI Users Meeting – 3/28/09

- Surviving the data deluge means computing in parallel
  - Good solutions for "easy" parallel problems, but gets fundamentally more difficult as dependencies get deeper
- Emerging technologies are a great start, but we need continued research integrating computational biology with research in HPC
  - A word of caution: new technologies are new

Acknowledgements

Advisor

Steven Salzberg

UMD Faculty

Mihai Pop, Art Delcher, Amitabh Varshney,
Carl Kingsford, Ben Shneiderman,
James Yorke, Jimmy Lin, Dan Sommer

CBCB Students

Adam Phillippy, Cole Trapnell,
Saket Navlakha, Ben Langmead,
James White, David Kelley

Thank You!

http://www.cbcb.umd.edu/~mschatz
@mike_schatz

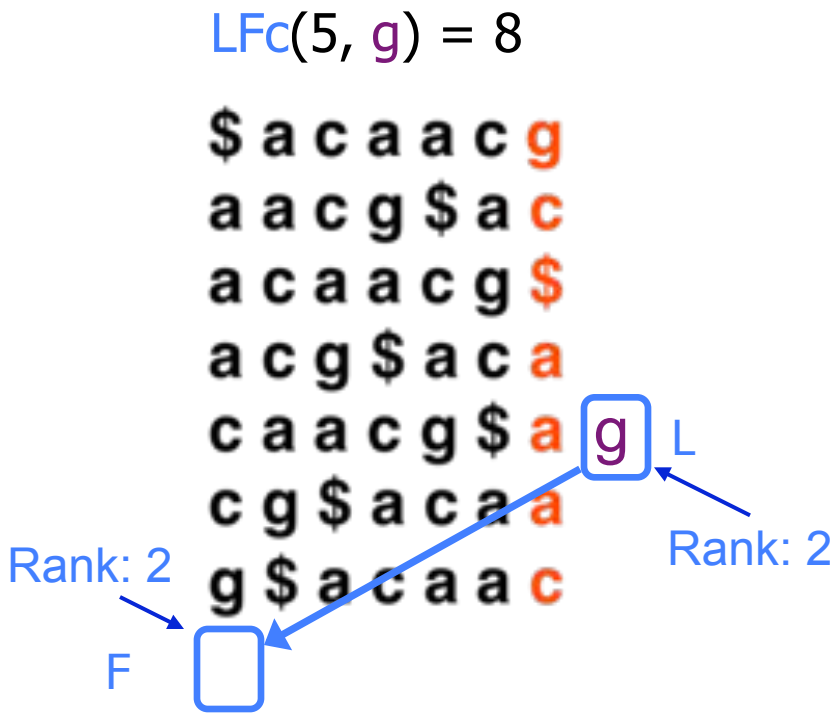
Burrows-Wheeler Transform

- Recreating T from BWT(T)
  - Start in the first row and apply **LF** repeatedly, accumulating predecessors along the way

BWT/Bowtie slides from Ben Langmead

Exact Matching

- **LFc**(r, c) does the same thing as **LF**(r) but it ignores r's actual final character and "pretends" it's c:



Exact Matching

- Start with a range, (**top**, **bot**) encompassing all rows and repeatedly apply **LFc**:

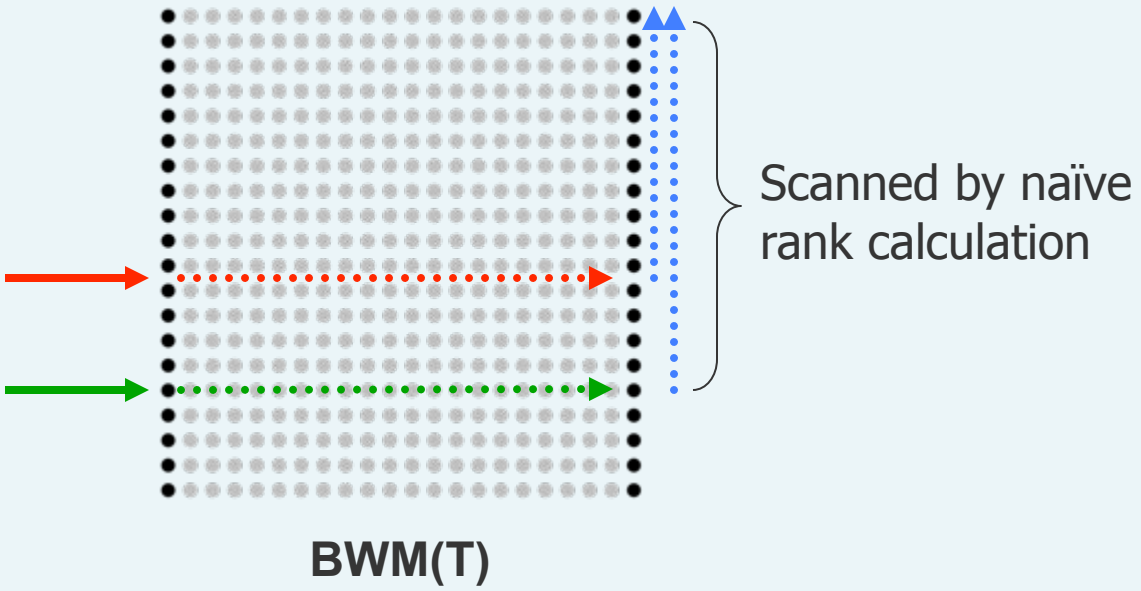
  top = **LFc**(**top**, **qc**); **bot** = **LFc**(**bot**, **qc**)

  qc = the next character to the left in the query

Ferragina P, Manzini G: Opportunistic data structures with applications. *FOCS. IEEE Computer Society; 2000.*

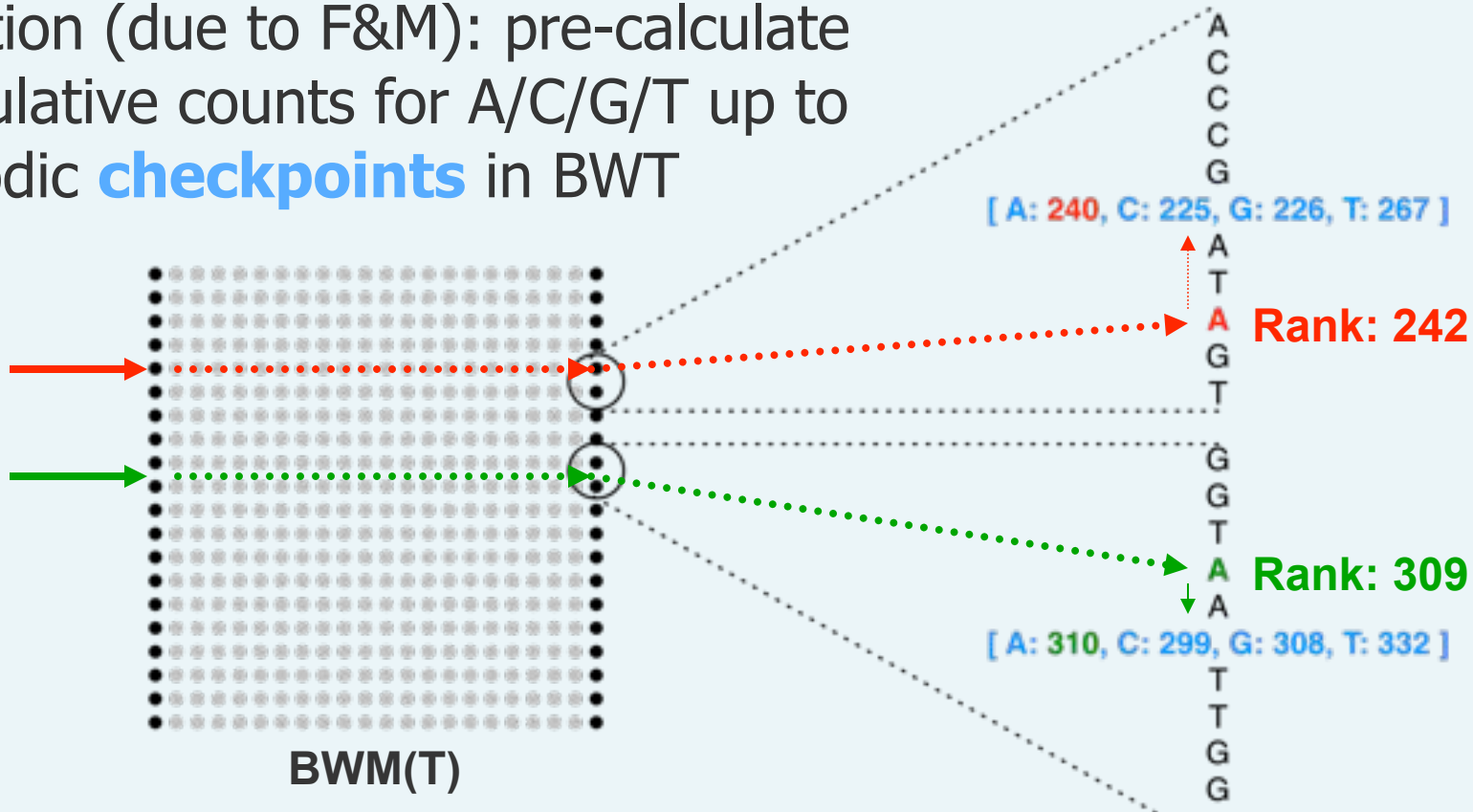
Checkpointing in FM Index

- **LF**(i, **qc**) must determine the *rank* of **qc** in row i
- Naïve way: count occurrences of **qc** in all previous rows
  - Linear in length of text – too slow



Checkpointing in FM Index

- Solution (due to F&M): pre-calculate cumulative counts for A/C/G/T up to periodic **checkpoints** in BWT



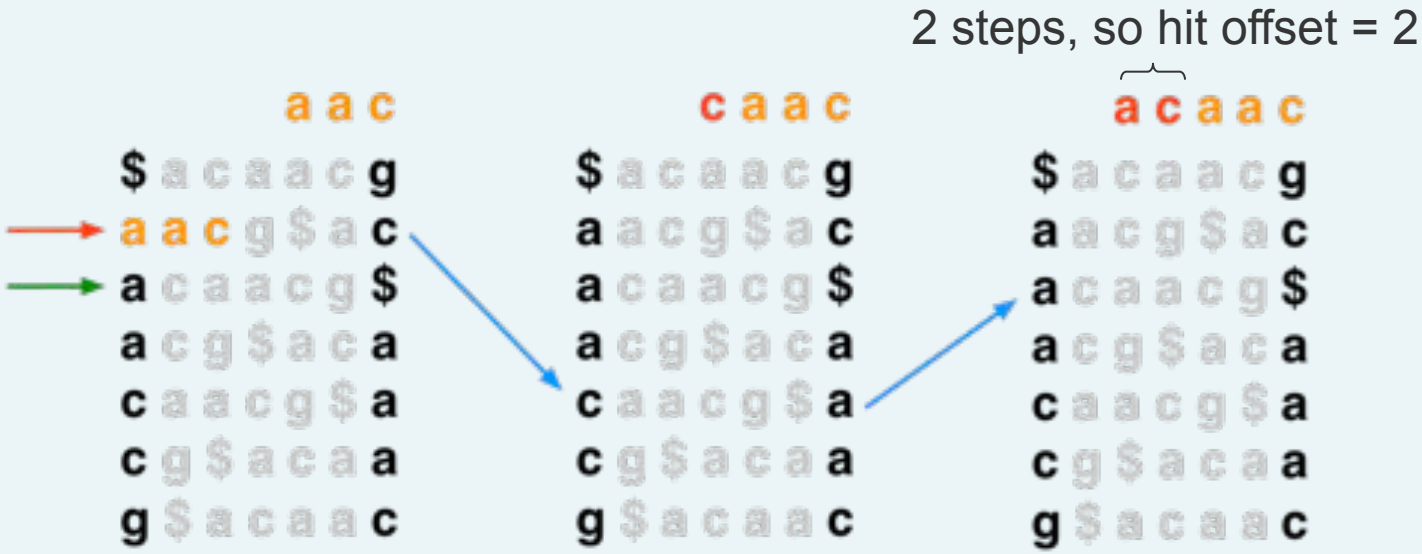
- **LF**(i, **qc**) is now constant time
  (if space between checkpoints is considered constant)

Rows to Reference Positions

- Once we know a row contains a legal alignment, how do we determine its position in the reference?

Rows to Reference Positions

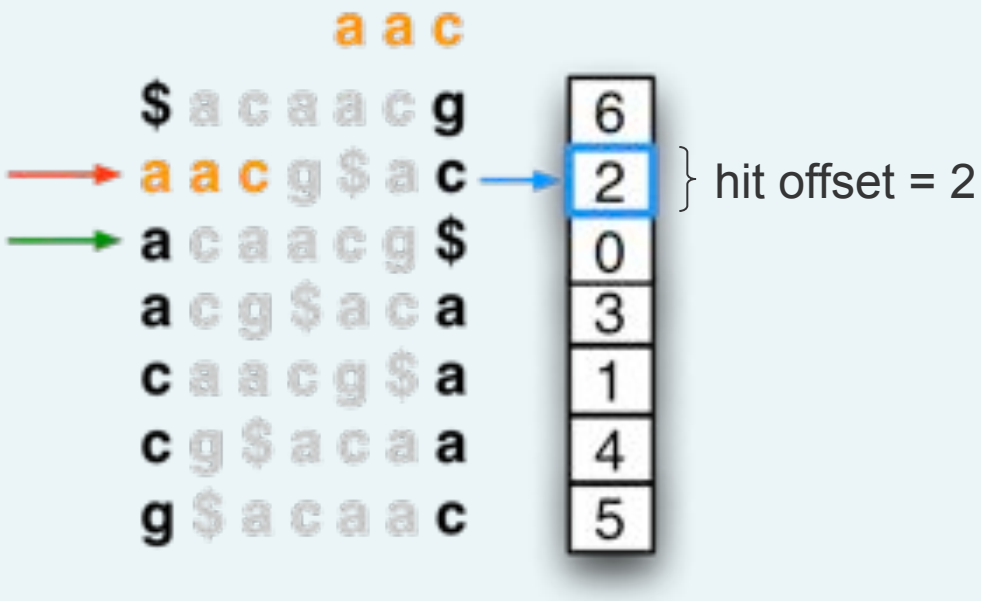
- Naïve solution 1: Use UNPERMUTE to walk back to the beginning of the text; number of steps = offset of hit



- Linear in length of text – too slow

Rows to Reference Positions

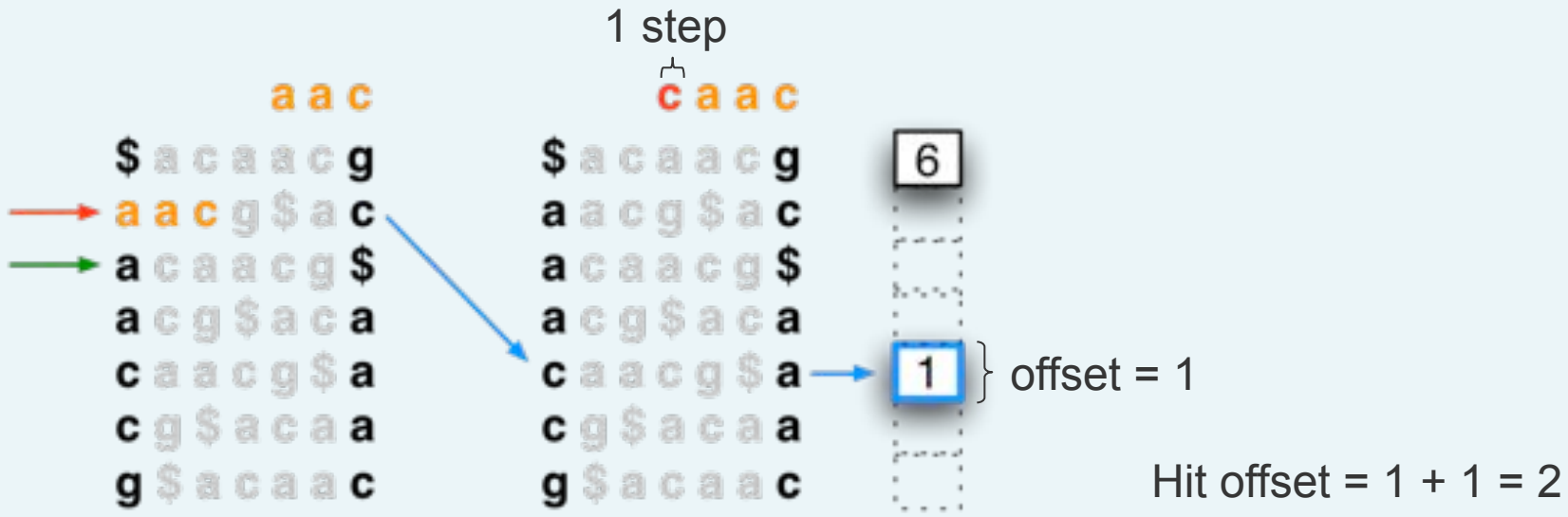
- Naïve solution 2: Keep pre-calculated offsets (the suffix array) in memory and do lookups



- Suffix array is ~12 GB for human – too big

Rows to Reference Positions

- Hybrid solution (due to F&M): Pre-calculate offsets for some "marked" rows; use UNPERMUTE to walk from the row of interest to next marked row to the left



- Bowtie marks every 32nd row by default (configurable)

FM Index is Small

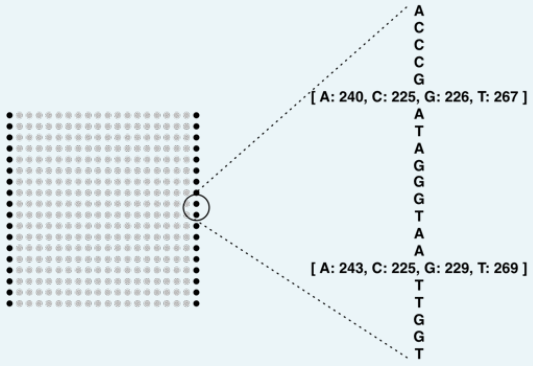
- Entire FM Index on DNA reference consists of:
  - BWT (same size as T)
  - Checkpoints (~15% size of T)
  - SA sample (~50% size of T)

Assuming 2-bit-per-base encoding and no compression, as in Bowtie

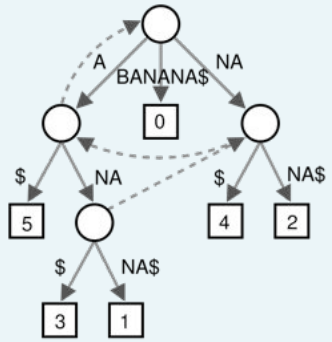
Assuming a 16-byte checkpoint every 448 characters, as in Bowtie

Assuming Bowtie defaults for suffix-array sampling rate, etc

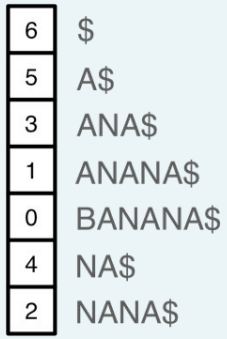
- Total: ~1.65x the size of T



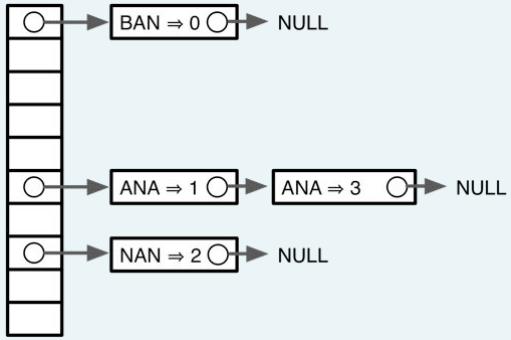
~1.65x



>45x



>15x



>15x