An Efficient Algorithm for Read Matching in DNA Databases

Dr. Yangjun Chen
Department of Applied Computer Science
University of Winnipeg

Outline

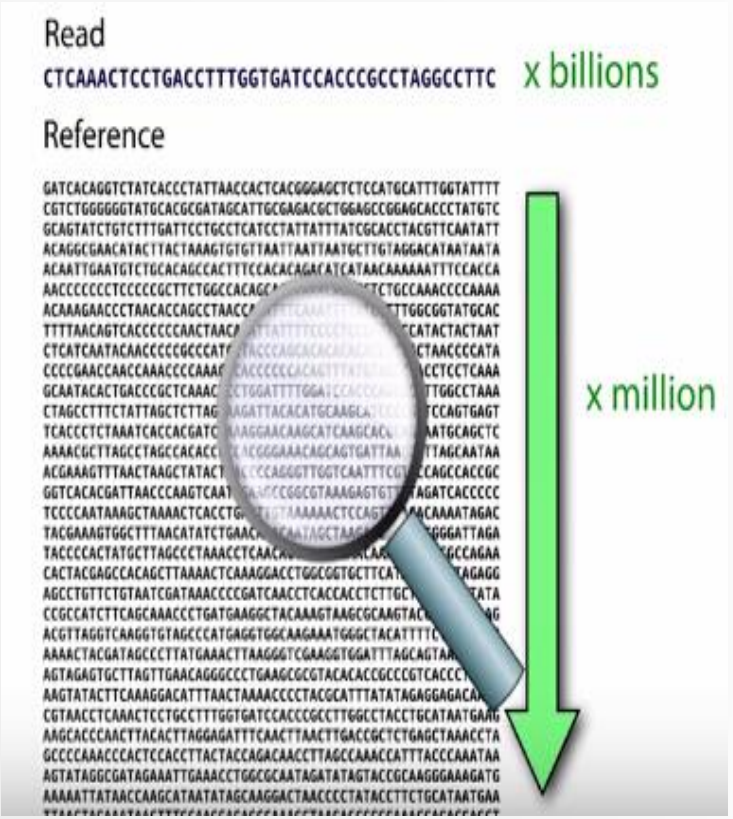
- Motivation
  - Statement of Problem
  - Related methods
- BWT Arrays – A Space-economic Index for String Matching
- Our Method
  - Combination of Tries and BWT Arrays
  - Multi-character searching
- Experiments
- Conclusion and Future Work

Statement of Problem

- Mapping massive reads (short DNA sequences) to reference sequences is the central computational problem for NGS (Next Generation Sequencing) data analysis.
  - Millions to billions short-reads are mapped to a reference genome sequence for statistic analysis.
  - Ability to produce short-reads has outpaced our ability to process them.

Short-Read Mapping

- Millions to billions short-reads need to be mapped.
- Reference genomes can be extremely large.
  - Human genome 3 billion bases.
  - Rat genome 2.9 billion bases.
- Short-reads may contain base errors compared to references. (Mismatch problem)

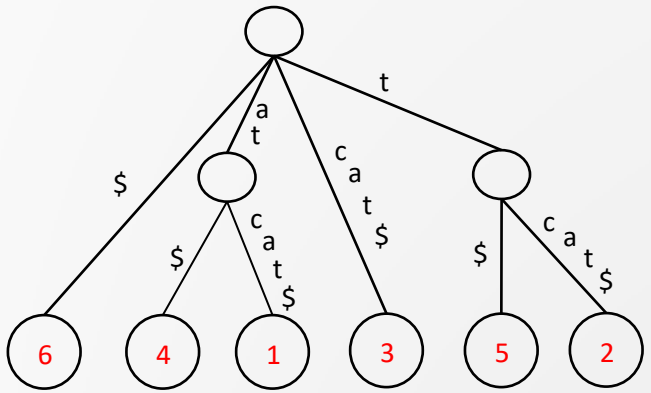


ACTACTGATC

CCTTGGACTACTGATCTTTAA

Related Methods

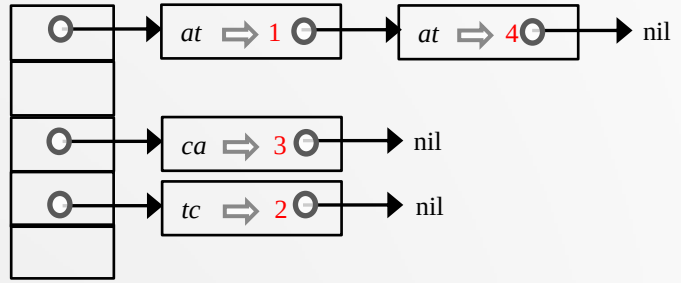
- Different kinds of indexes: suffix trees, suffix arrays, hash tables
- Example: reference sequence = *atcat$*



Suffix tree

i	Position	Suffix
1	6	$
2	4	*at*$
3	1	*atcat*$
4	3	*cat*$
5	5	*t*$
6	2	*tcat*$

Suffix array



Hash Table

- **Indices can be big. For human: suffix tree > 50 Gb, suffix array > 12 Gb, hash table > 12 Gb.**

BWT-Index

- **Burrows-Wheeler Transform (*BWT*)**
- $s = a_1c_1a_2g_1a_3c_2a_4\$$

$a_1\ c_1\ a_2\ g_1\ a_3\ c_2\ a_4\ \$$
$c_1\ a_2\ g_1\ a_3\ c_2\ a_4\ \$\ a_1$
$a_2\ g_1\ a_3\ c_2\ a_4\ \$\ a_1\ c_1$
$g_1\ a_3\ c_2\ a_4\ \$\ a_1\ c_1\ a_2$
$a_3\ c_2\ a_4\ \$\ a_1\ c_1\ a_2\ g_1$
$c_2\ a_4\ \$\ a_1\ c_1\ a_2\ g_1\ a_3$
$a_4\ \$\ a_1\ c_1\ a_2\ g_1\ a_3\ c_2$
$\$\ a_1\ c_1\ a_2\ g_1\ a_3\ c_2\ a_4$

Rank correspondence:

rank: 3

rk_F	F	L	rk_L
-	$\$\ a_1\ c_1\ a_2\ g_1\ a_3\ c_2$	a_4	1
1	$a_4\ \$\ a_1\ c_1\ a_2\ g_1\ a_3$	c_2	1
2	$a_3\ c_2\ a_4\ \$\ a_1\ c_1\ a_2$	g_1	1
3	$a_1\ c_1\ a_2\ g_1\ a_3\ c_2\ a_4$	$\$$	-
4	$a_2\ g_1\ a_3\ c_2\ a_4\ \$\ a_1$	c_1	2
1	$c_2\ a_4\ \$\ a_1\ c_1\ a_2\ g_1$	a_3	2
2	$c_1\ a_2\ g_1\ a_3\ c_2\ a_4\ \$$	a_1	3
1	$g_1\ a_3\ c_2\ a_4\ \$\ a_1\ c_1$	a_2	4

rank: 3

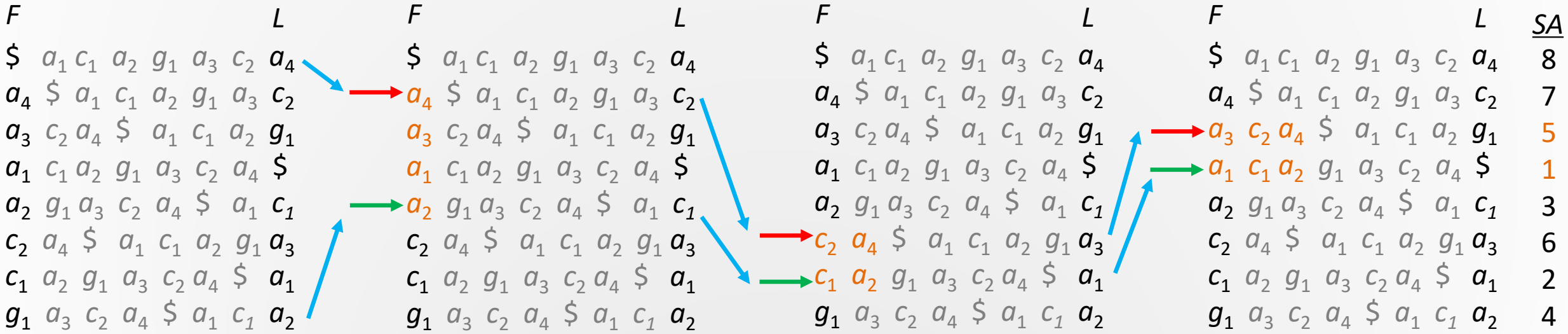
BWT construction:

$$\begin{cases} L[i] = \$, & \text{if } SA[i] = 0; \\ L[i] = s[SA[i] - 1], & \text{otherwise.} \end{cases}$$

***SA*[...] – suffix array**

Backward Search of BWT-Index

- $s = a_1c_1a_2g_1a_3c_2a_4\$$
- Search $p = aca$



rankAll

- Arrange $|\Sigma|$ arrays each for a character $\alpha \in \Sigma$ such that $A_\alpha[i]$ (the ith entry in the array for α) is the number of appearances of α within $L[1 .. i]$.
- Instead of scanning a certain segment $L[x .. y]$ ($x \leq y$) to find a subrange for a certain $\alpha \in \Sigma$, we can simply look up A_α to see whether $A_\alpha[x - 1] = A_\alpha[y]$. If it is the case, then α does not occur in $L[x .. y]$. Otherwise, $[\alpha[x - 1] + 1, \alpha[y]]$ should be the found range.

Example

To find the first and the last appearance of c in $L[2 .. 5]$, we only need to find $c[2 - 1] = c[1] = 0$ and $c[5] = 2$. So the corresponding range is $[c[2 - 1] + 1, c[5]] = [1, 2]$.

F	L
\$	a_4
a_4	c_2
a_3	g_1
a_1	\$
a_2	c_1
c_2	a_3
c_1	a_1
g_1	a_2

$A_\$$	A_a	A_c	A_g	A_t
0	1	0	0	0
0	1	1	0	0
0	1	1	1	0
1	1	1	1	0
1	1	2	1	0
1	2	2	1	0
1	3	2	1	0
1	4	2	1	0

Reduce *rankAll*-Index Size

- F ranks: $F_\varepsilon = <\alpha; x_\alpha, y_\alpha>$
- BWT array: L
- Reduced appearance array: A_α with bucket size β.
- Reduced suffix array: SA^* with bucket size γ.

$$top = F(x_\alpha) + A_\alpha[\lfloor (top\text{-}1) / \beta \rfloor] + r + 1$$

$$bot = F(x_\alpha) + A_\alpha[\lfloor bot / \beta \rfloor] + r'$$

r is the number of α's appearances within $L[\lfloor (top - 1) / \beta \rfloor \beta \ldots top - 1]$

r' is the number of α's appearances within $L[\lfloor bot / \beta \rfloor \beta \ldots bot]$

i	F	L	rk_L	SA
1	\$	a_4	1	8
2	a_4	c_2	1	7
3	a_3	g_1	1	5
4	a_1	\$	-	1
5	a_2	c_1	2	3
6	c_2	a_3	2	6
7	c_1	a_1	3	2
8	g_1	a_2	4	4

$F_\varepsilon = <\alpha; x_\alpha, y_\alpha>$

$F_\$ = <\$; 1, 1>$
$F_a = <a; 2, 5>$
$F_c = <c; 6, 7>$
$F_g = <g; 8, 8>$

\+

L
a_4
c_2
g_1
\$
c_1
a_3
a_1
a_2

\+

$A_\$$	A_a	A_c	A_g	A_t
0	1	0	0	0
0	1	1	0	0
0	1	1	1	0
1	1	1	1	0
1	1	2	1	0
1	2	2	1	0
1	3	2	1	0
1	4	2	1	0

\+

SA^*
8
7
5
1
3
6
2
4

Our Approach

- By BWT-arrays, reads are searched one by one.
- We consider all reads as a whole to avoid recalculation.
  - When total amount of reads is large, many reads share common prefixes.
  - Search of same subsequences will result in same rank segment using BWT-index.

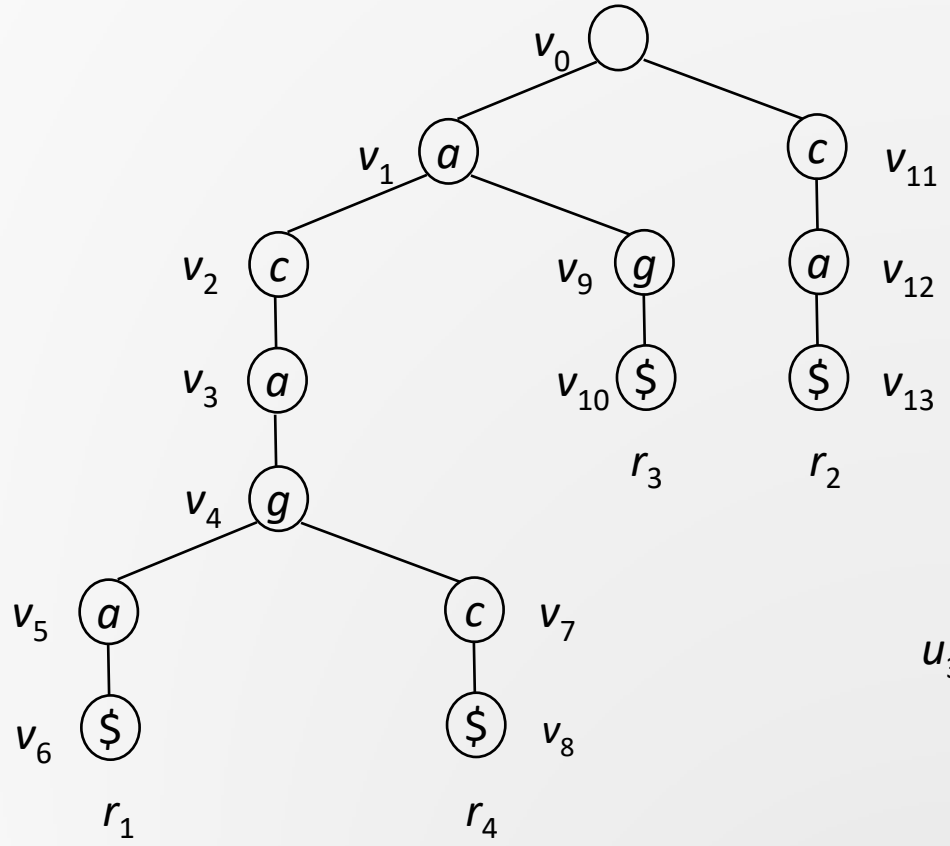
Methodology

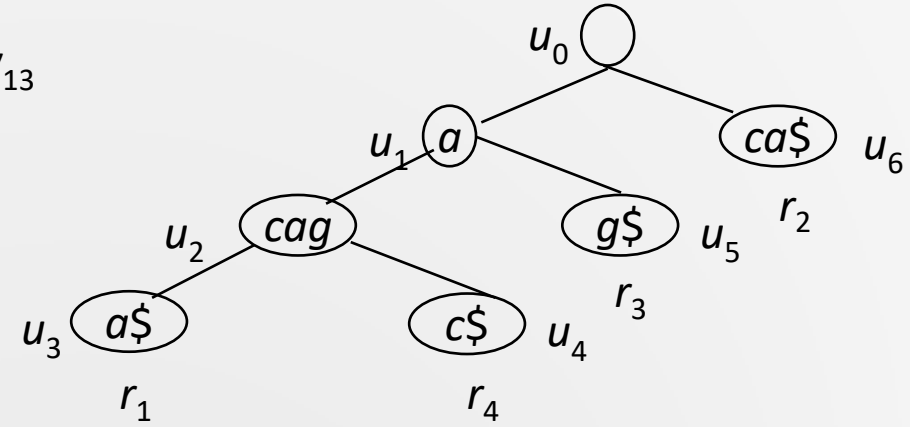
- Arrange a set of reads into a ***trie*** structure.
- Search the ***trie*** against BWT arrays created for a reference genome.
- **Multi-character checking** when scanning a segment of *L* in BWT index to reduce the frequency of accessing *L*.

Trie Construction

- **Arrange all reads into a trie structure**

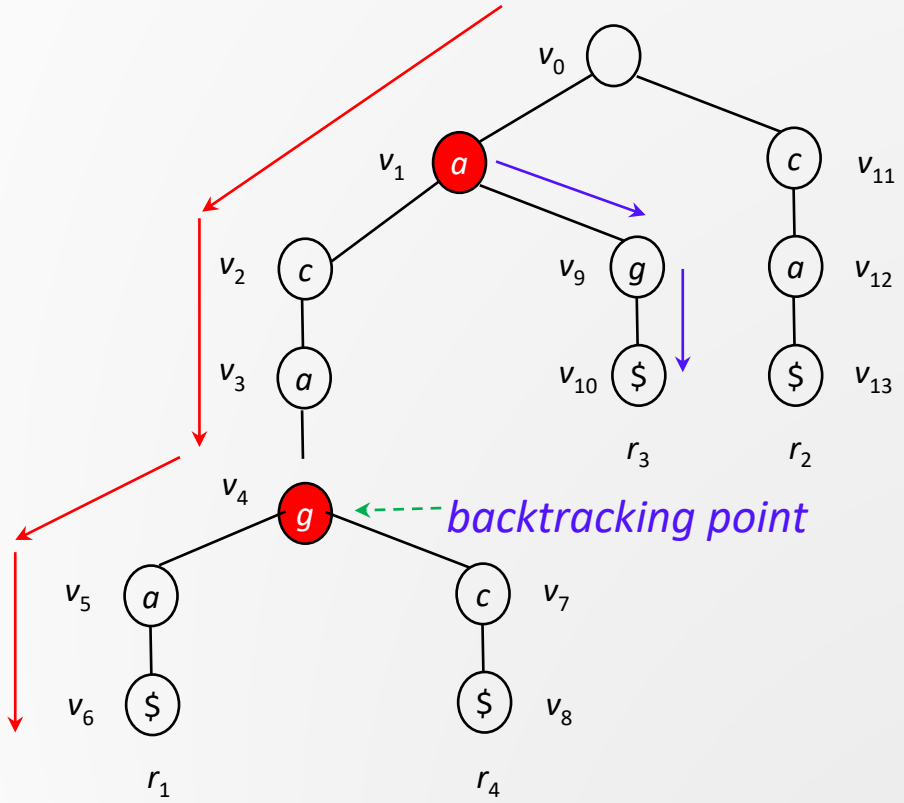
ID	Read sequence
r_1	*acaga*
r_2	*ca*
r_3	*ag*
r_4	*acagc*

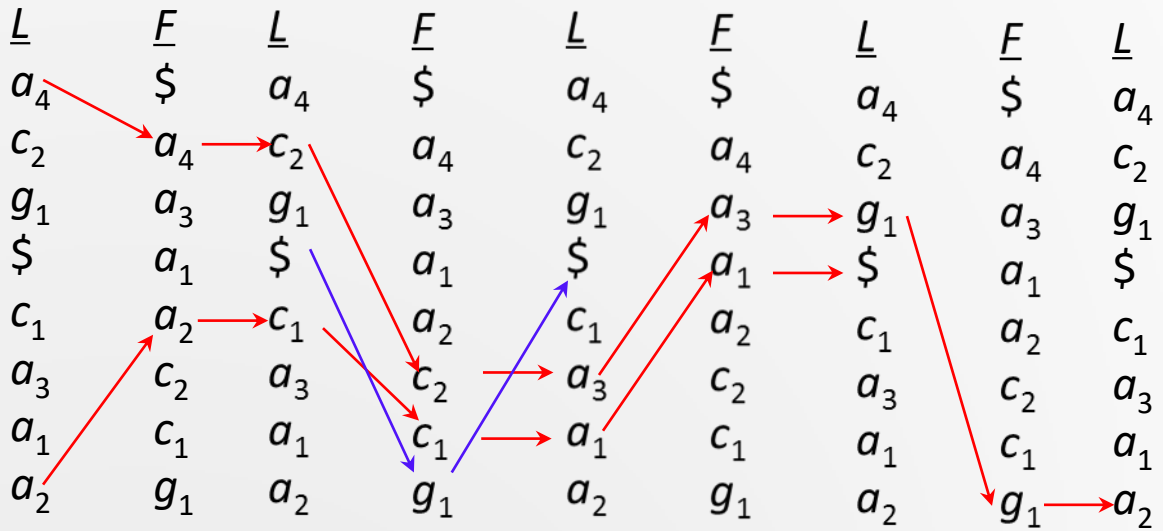




Trie Searching against BWT Array

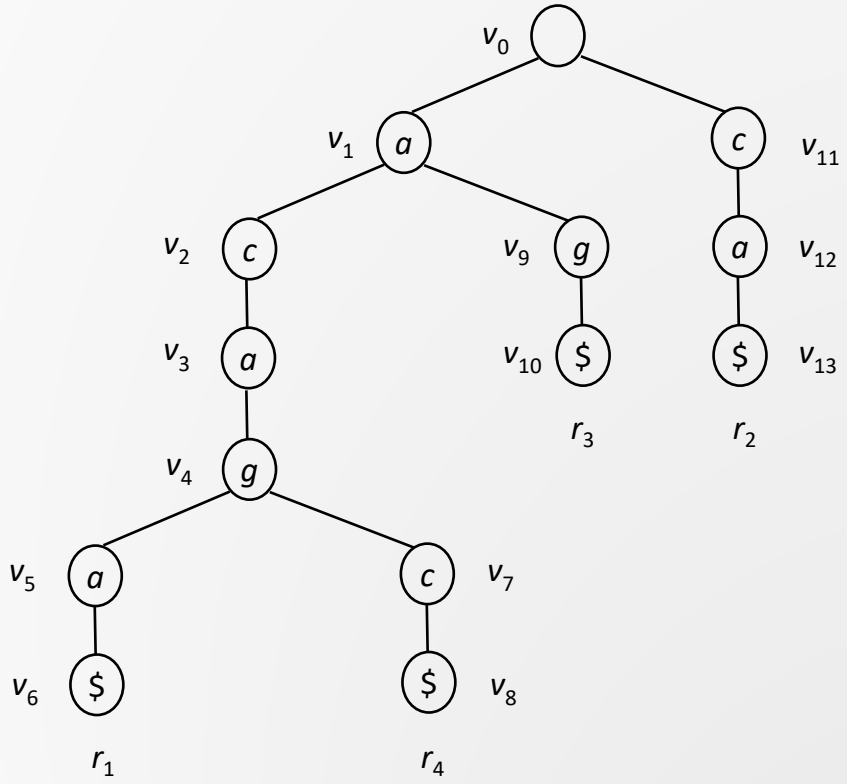
- Search a trie structure in the depth-first manner



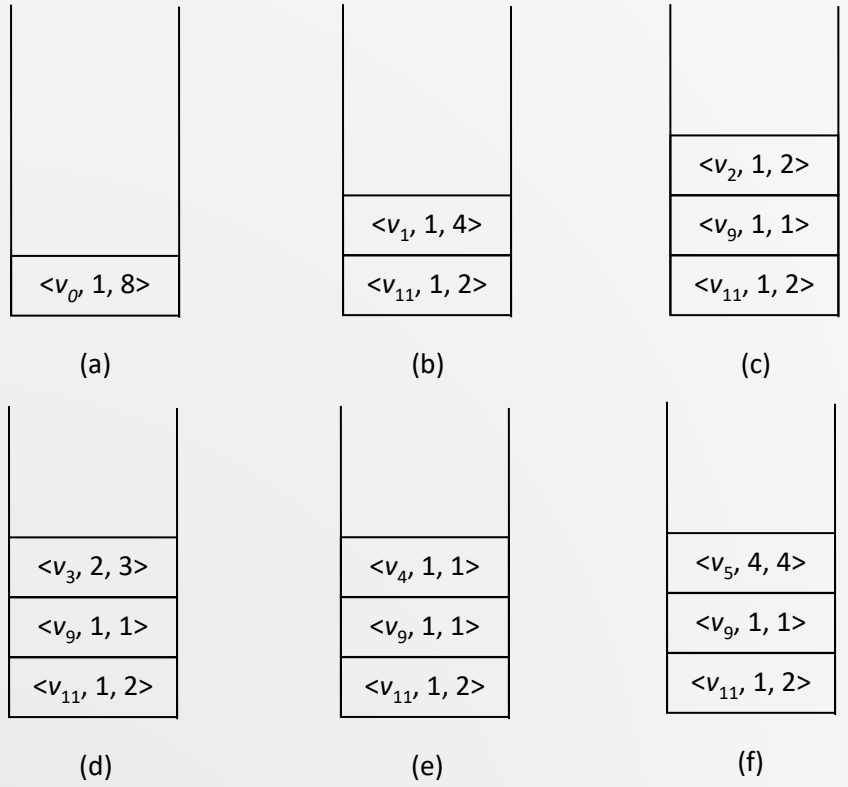


Simultaneously Search Trie and BWT-Index

- Search the trie against BWT-index created for a reference genome
- Keep intermediate ranks in a stack

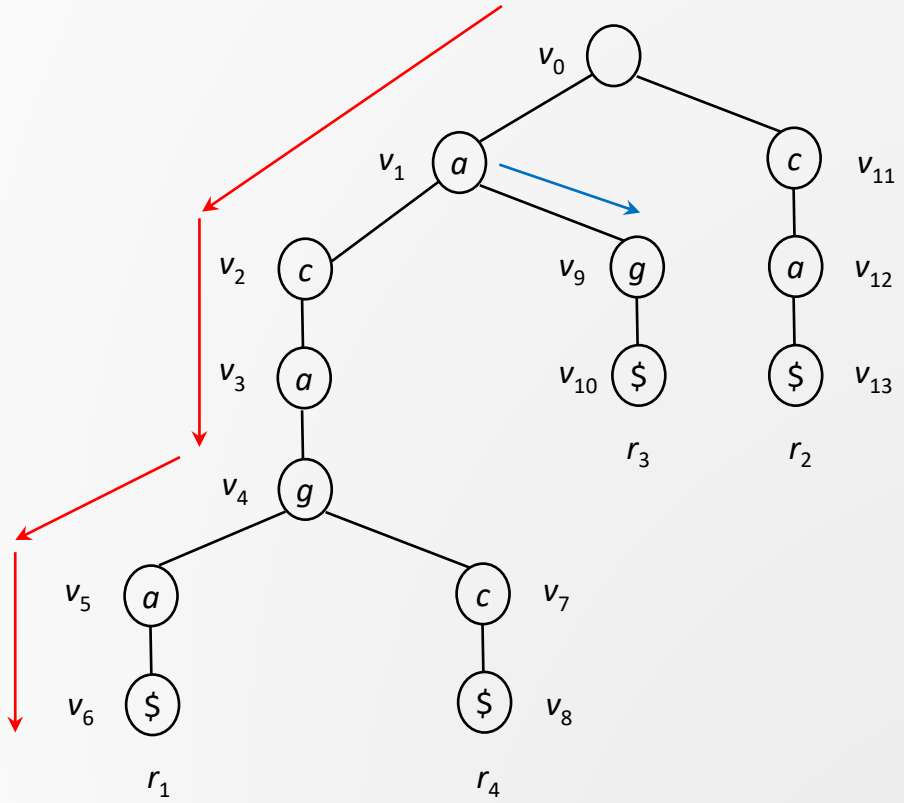


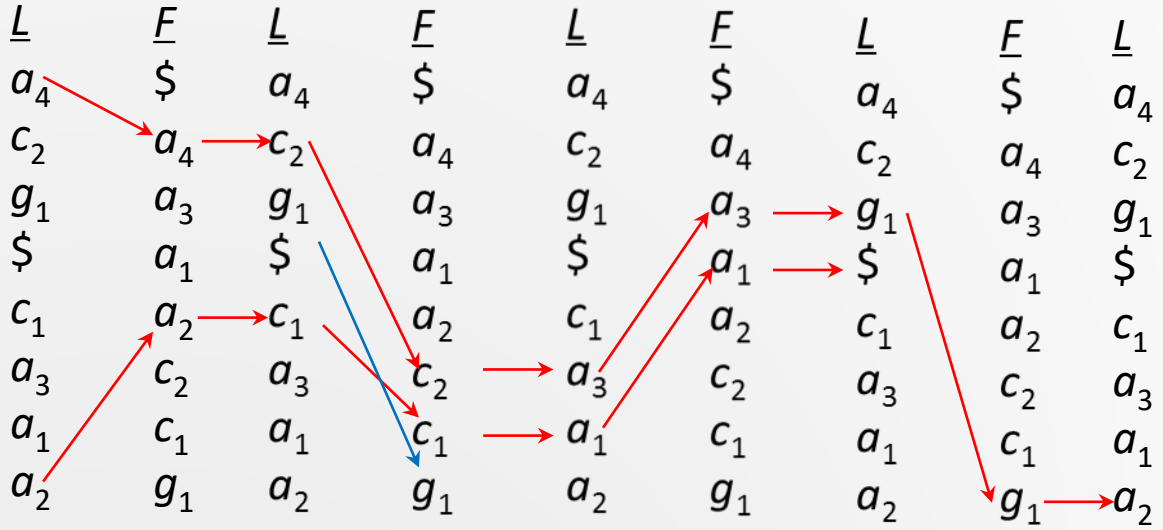
Stack:



Multiple Character Searching

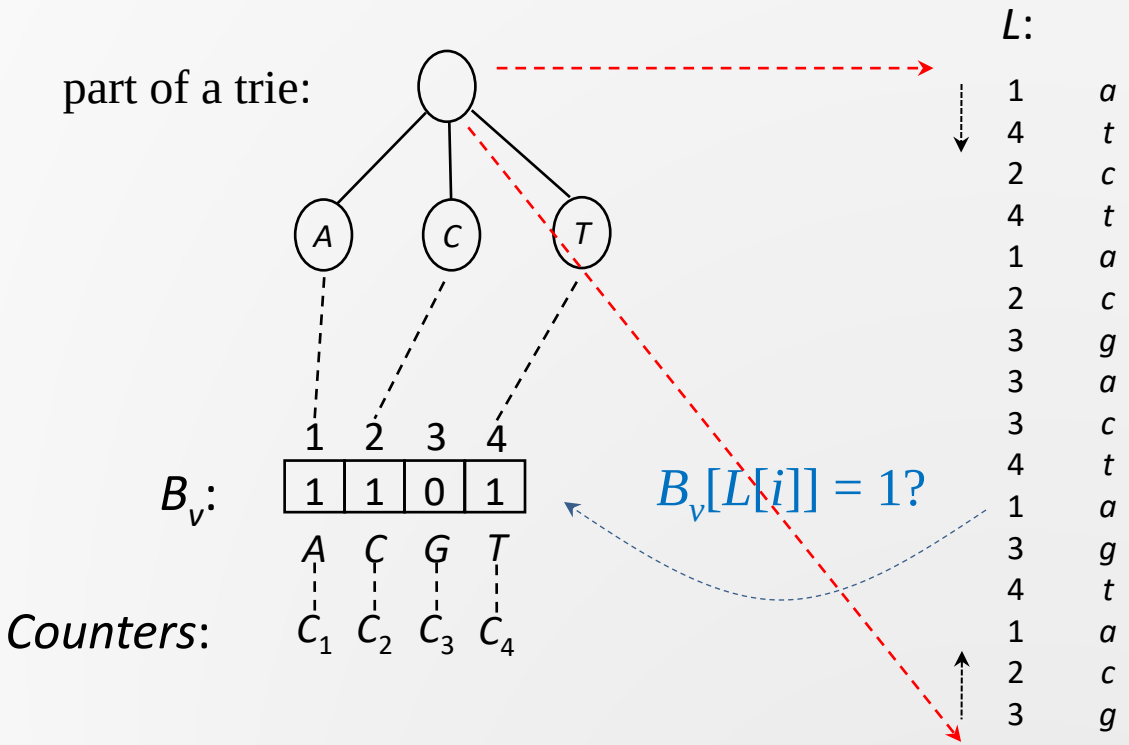
- Search a trie structure





Multi-Character Checking

- **Multi-character checking when scanning a segment of *L* in BWT-index.**



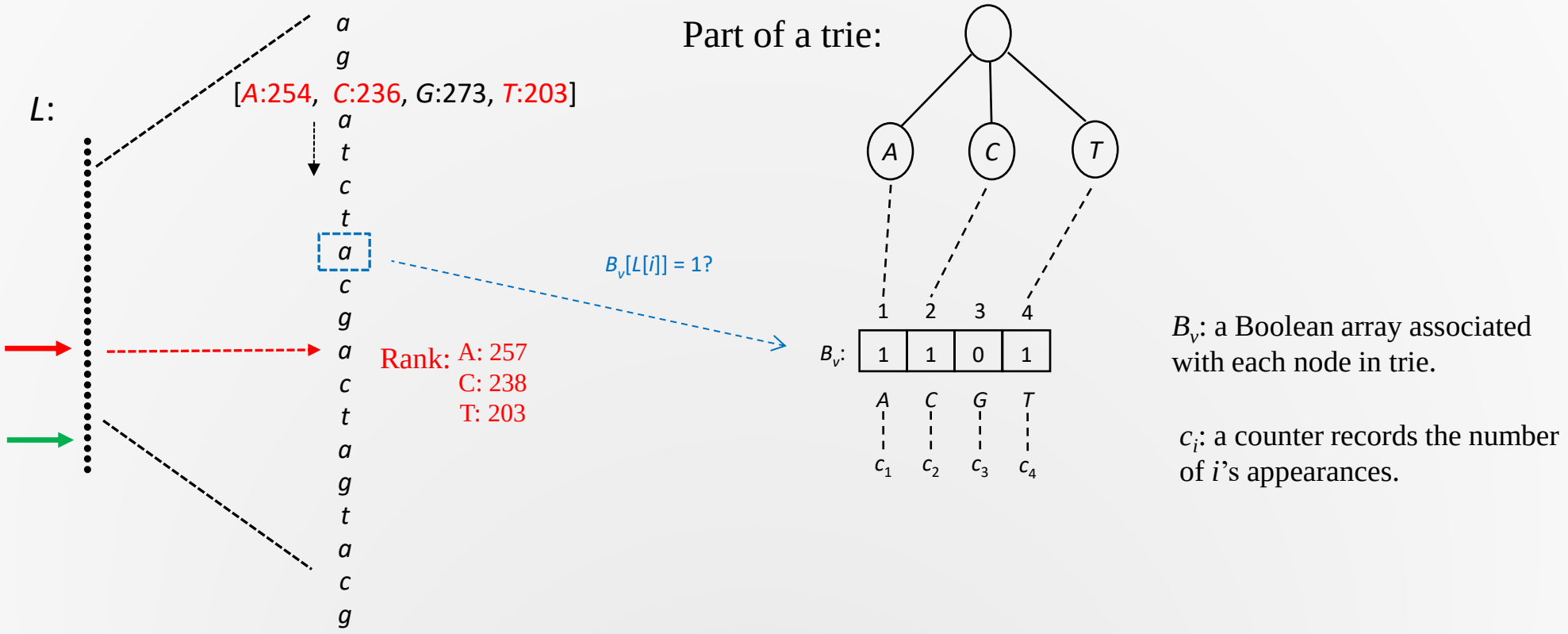
B_v: a Boolean array associated with each node in a trie.

C_i: a counter records the number of *i*'s appearances.

If $B_v[L[i]] = 1$ then $C_{L[i]} := C_{L[i]} + 1$

Multi-Character Checking

- Multi-character checking when scanning a segment of L in FM-index.



B_v: a Boolean array associated with each node in trie.

c_i: a counter records the number of i's appearances.

Experiments

- Compare 5 different approaches
  - *Burrows Wheeler Transformation* (BWT for short),
  - *Suffix tree based* (Suffix for short),
  - *Hash table based* (Hash for short),
  - *Trie-BWT* (tBWT for short, discussed in this paper),
  - *Improved Trie-BWT* (itBWT for short, discussed in this paper).

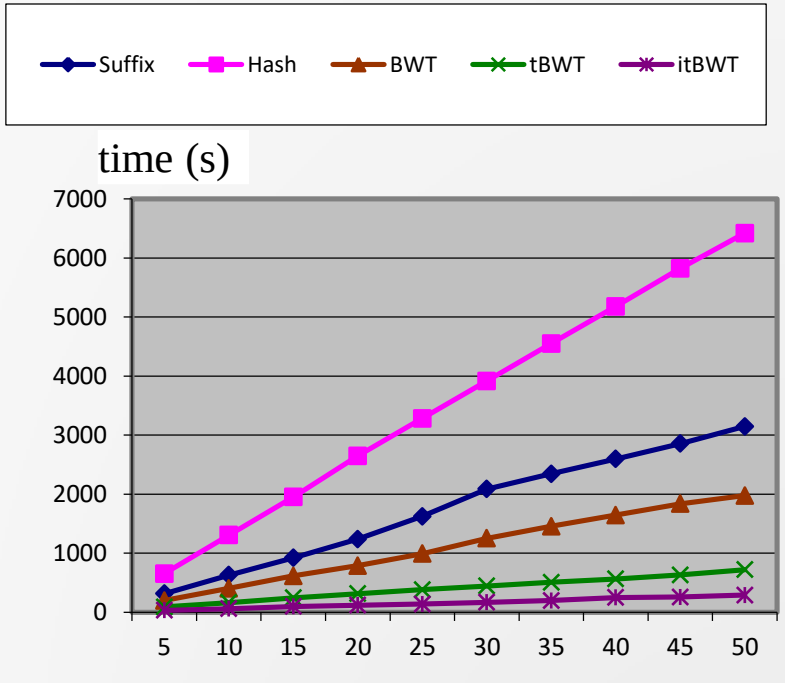
Experiments

TABLE I. CHARACTERISTICS OF GENOMES

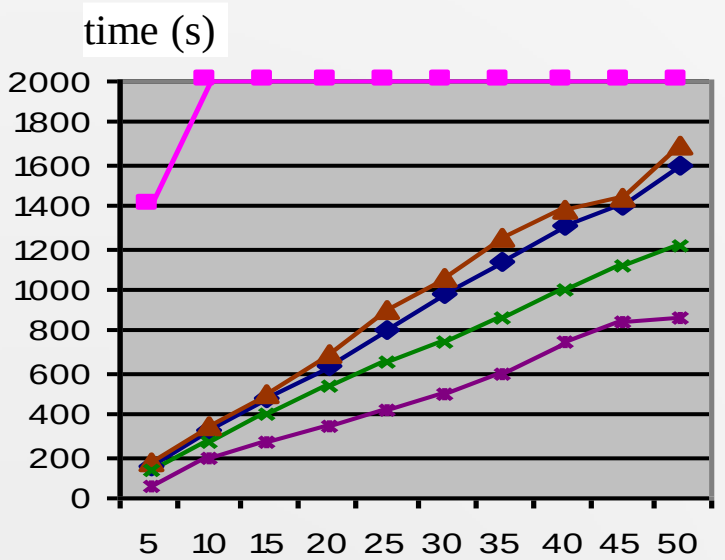
Genomes	Genome sizes (bp)
Rat chr1 (Rnor_6.0)	290,094,217
C. merolae (ASM9120v1)	16,728,967
C. elegans (WBcel235)	103,022,290
Zebra fish (GRCz10)	1,464,443,456
Rat (Rnor_6.0)	2,909,701,677

Tests with Synthetic Data

- **Tests with varying amount of reads (over Rat chr1)**



amount of reads with length 100 pbs (million)

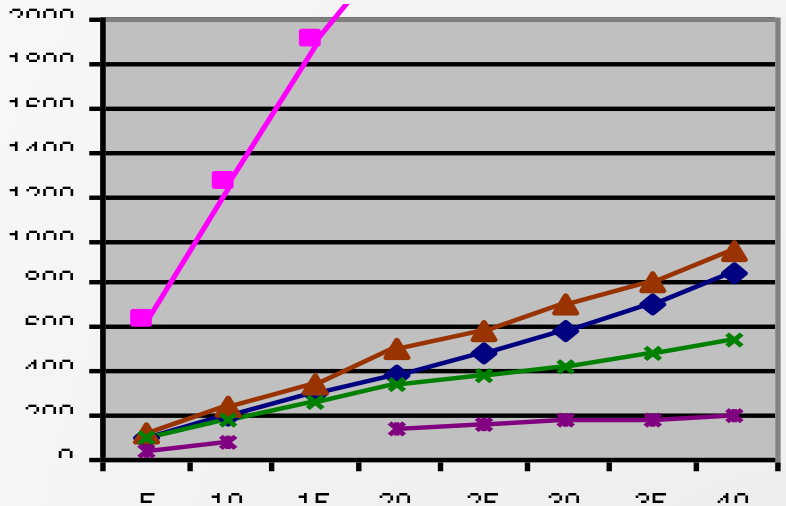


amount of reads with length 50 pbs (million)

Tests with Synthetic Data

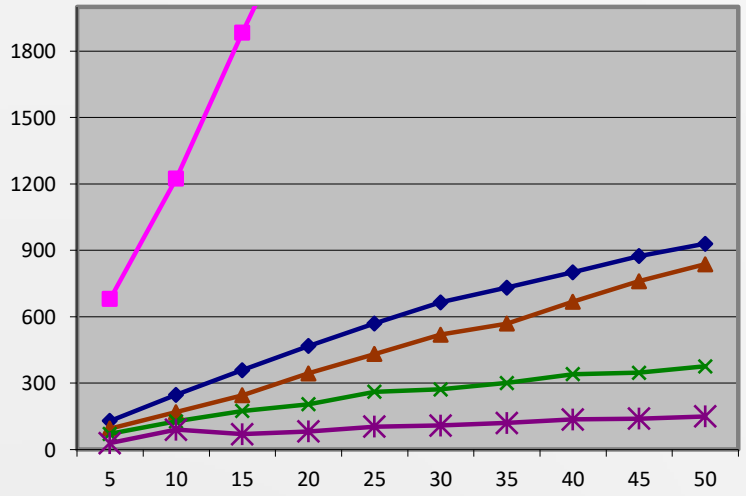
- **Tests with varying amount of reads (over *C. merolae*)**

time (s)



amount of reads with length 50 pbs (million)

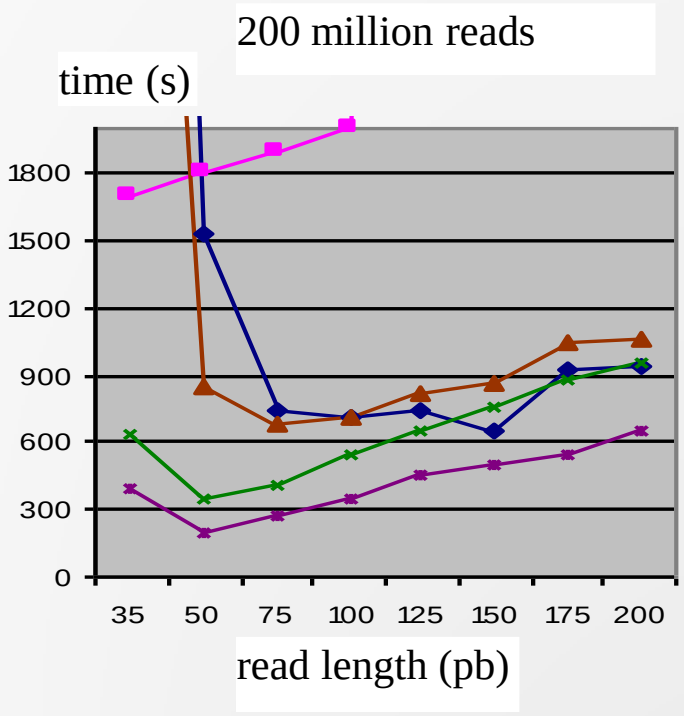
time (s)

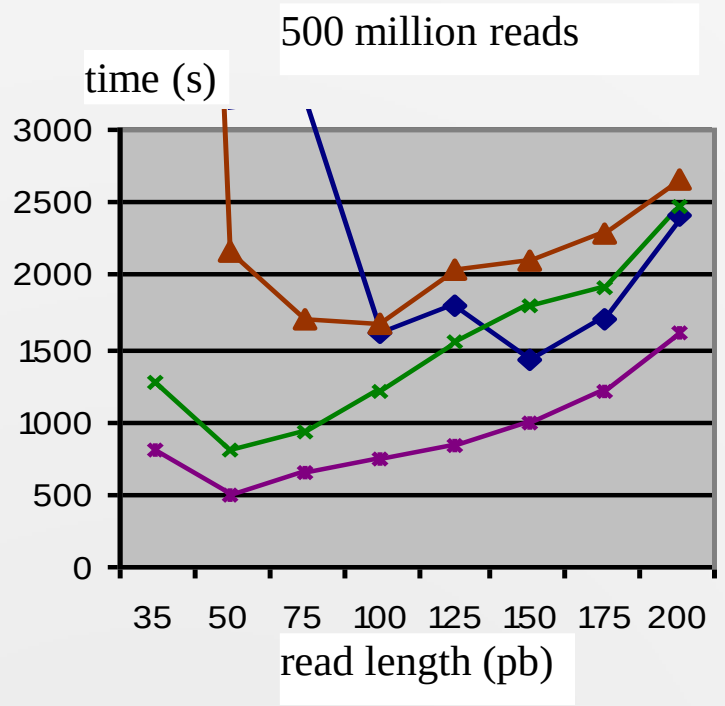


amount of reads with length 100 pbs (million)

Tests with Synthetic Data

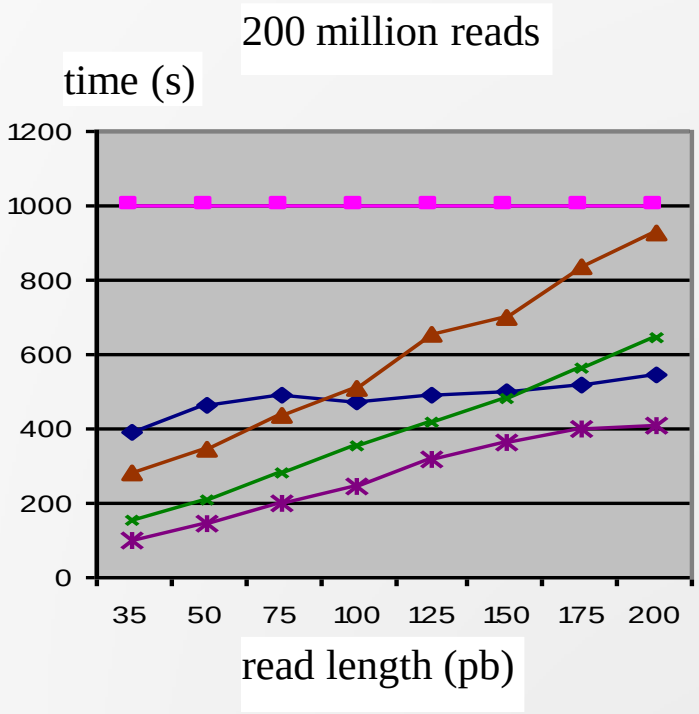
- Tests with varying length of reads (OVER **Rat chr1**)

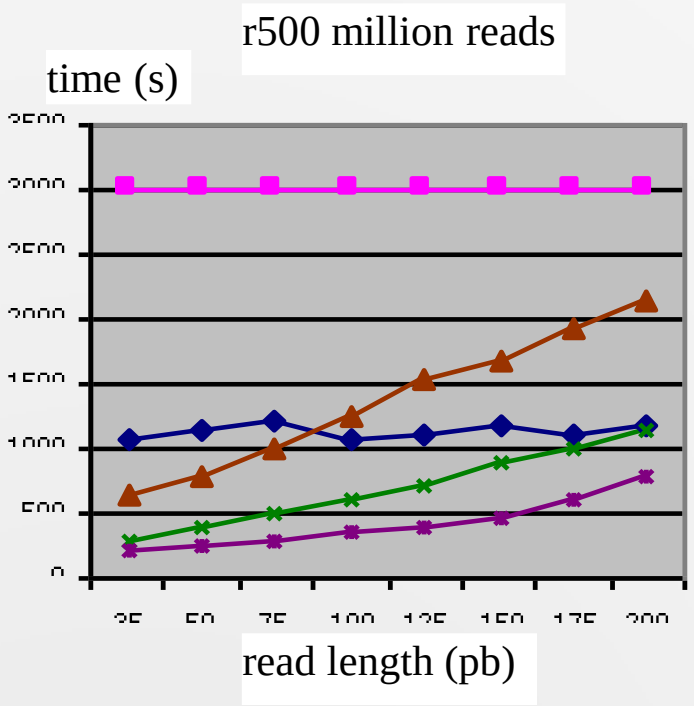




Tests with Synthetic Data

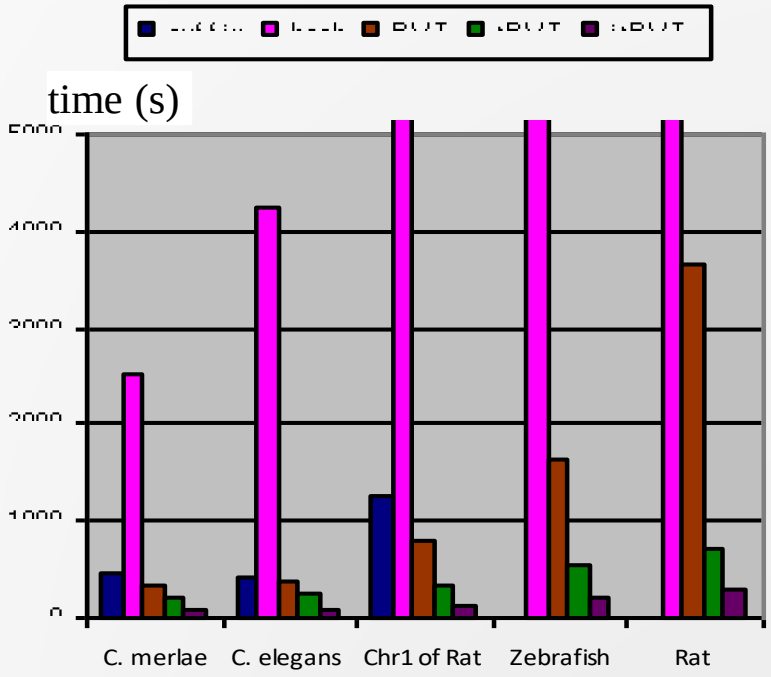
- **Tests with varying length of reads (OVER C. merlae)**

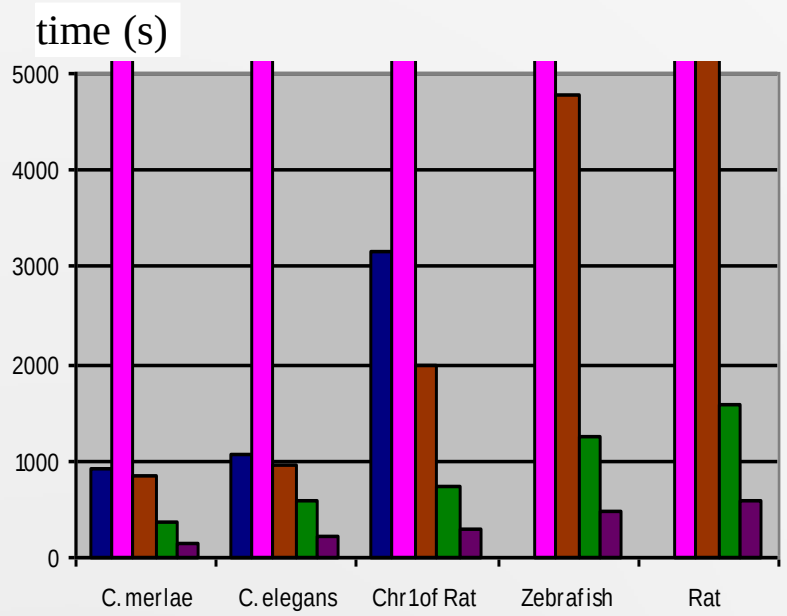




Tests with Synthetic Data

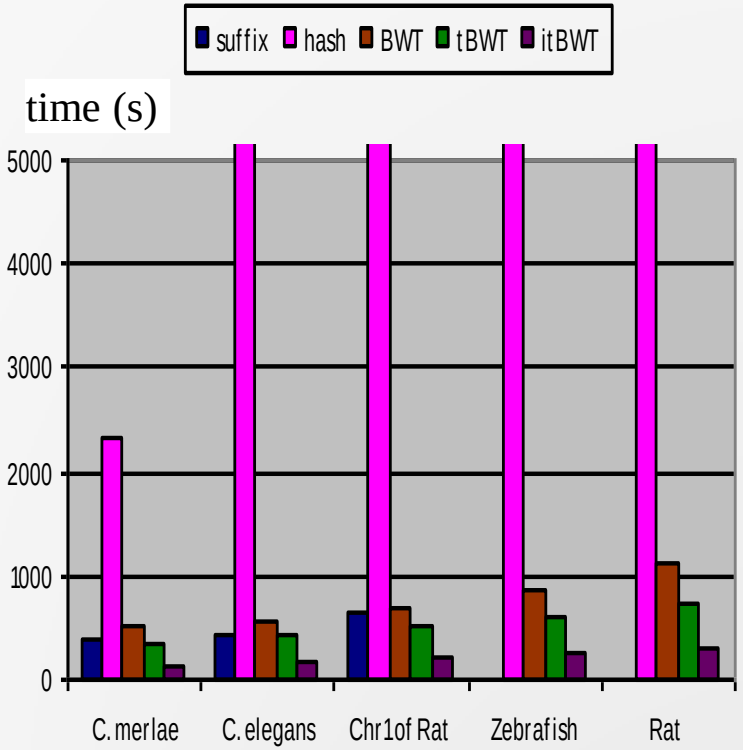
- Tests with varying sizes of genome (20 million and 50 million reads of 50 bps)

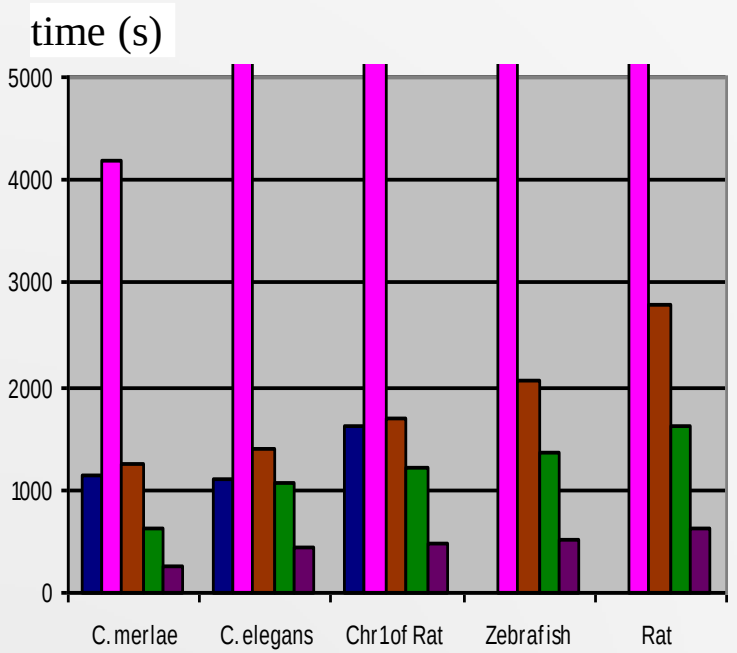




Tests with Synthetic Data

- Tests with varying sizes of genome (20 million and 50 million reads of 100 bps)

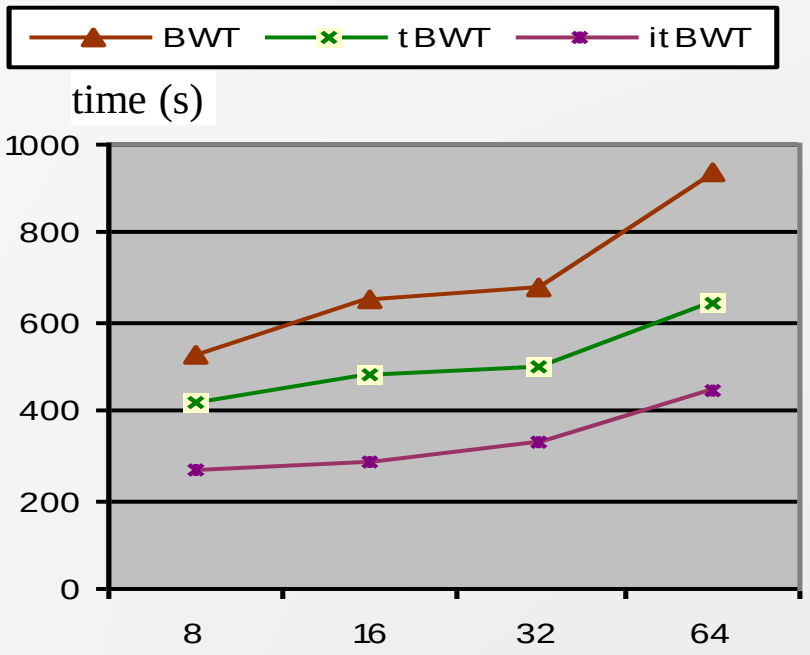




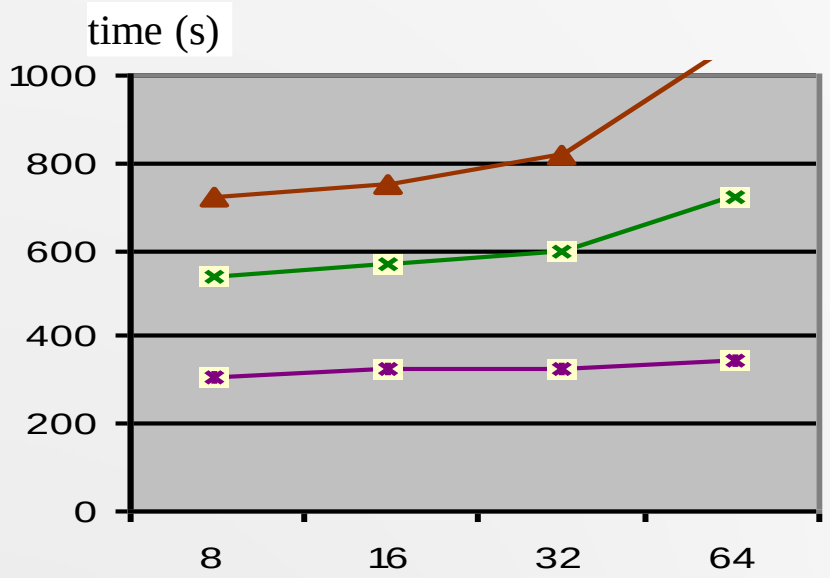
Tests with Synthetic Data

- **Tests on compression factors (20 million reads with 100 bps in length)**

Suffix array compression factors set to be 16, 64.



rankALL compression factors

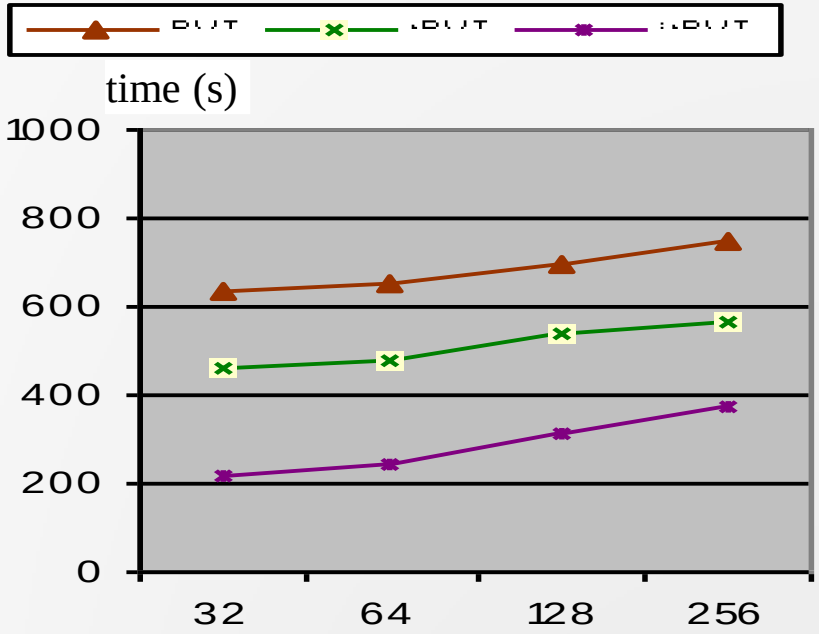


rankALL compression factors

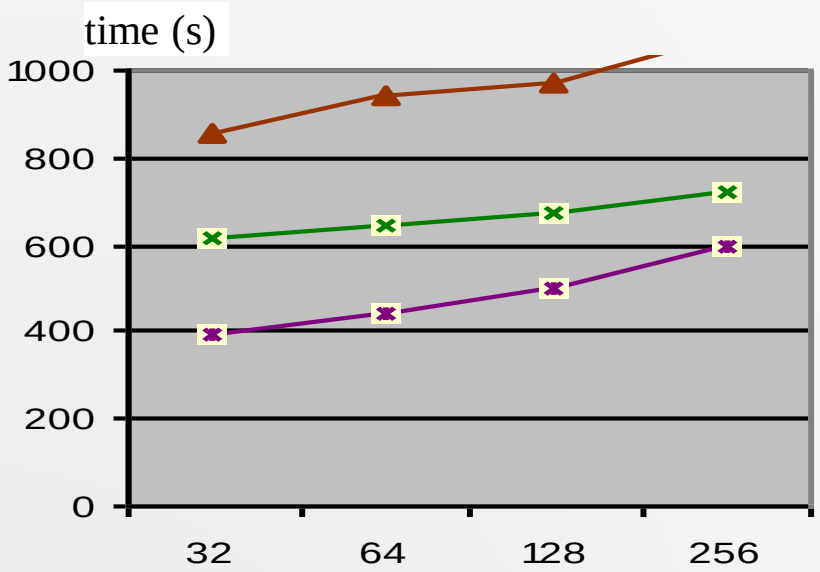
Tests with Synthetic Data

- **Tests on compression factors (20 million reads with 100 bps in length)**

Suffix array compression factors set to be 64, 256.



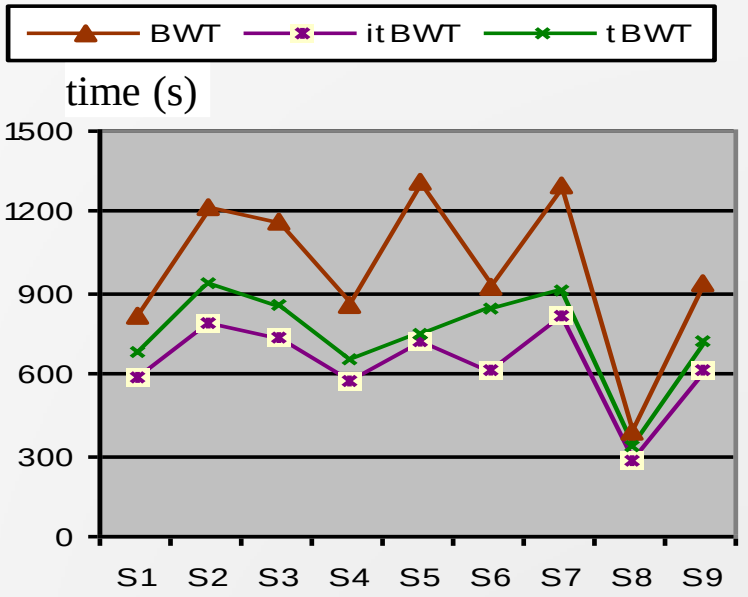
rankALL compression factor



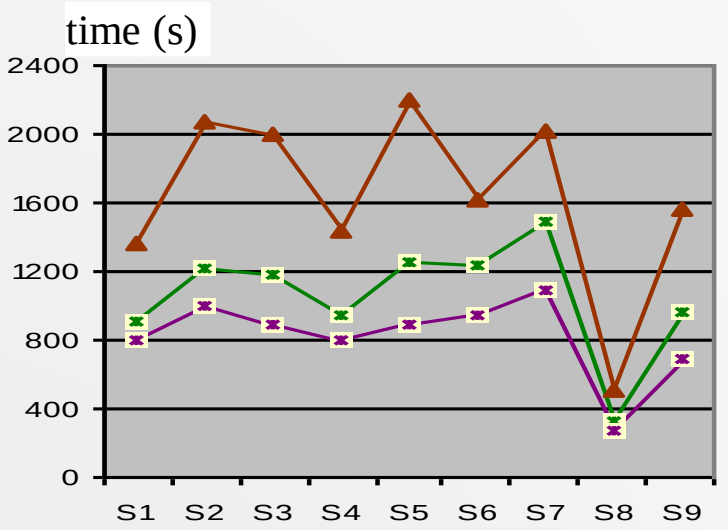
rankALL compression factor

Tests with Real Data

- 500 million single reads produced by Illumina from a rat sample.
- Length of these reads: 36 bps and 100 bps after trimming using Trimmomatic .
- The reads divided into 9 samples with different amount: between 20 and 75 million.
- mapping the 9 samples back to rat genomes



mapping the 9 samples back to rat genome of ENSEMBL release 79



mapping the 9 samples back to the Rat *transcriptome*

Conclusion and future work

- **Main contribution**
  - Combination of trie and BWT indexes
  - Multi-character checking
  - Extensive tests
- **Future work**
  - Adapt our matching algorithm for protein sequences
  - String matching with *k* Mismatch

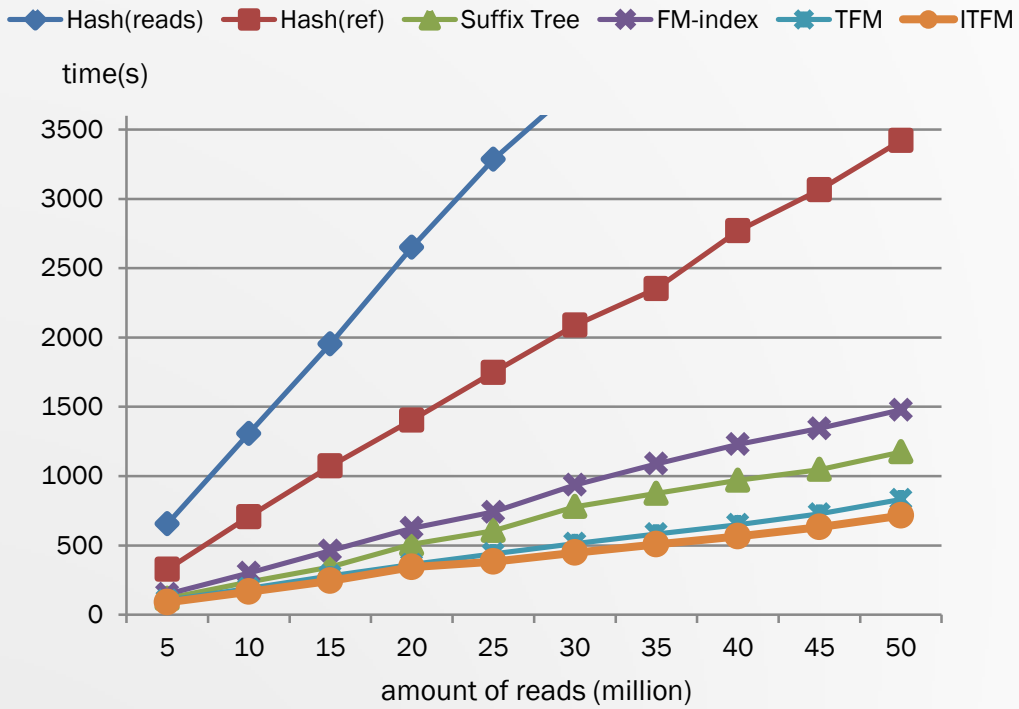
Thank you!

Varying Read Amount

- Genome size = chromosome 1 of Rat genome, 290,094,217 bp.
- Read length = 50 bp

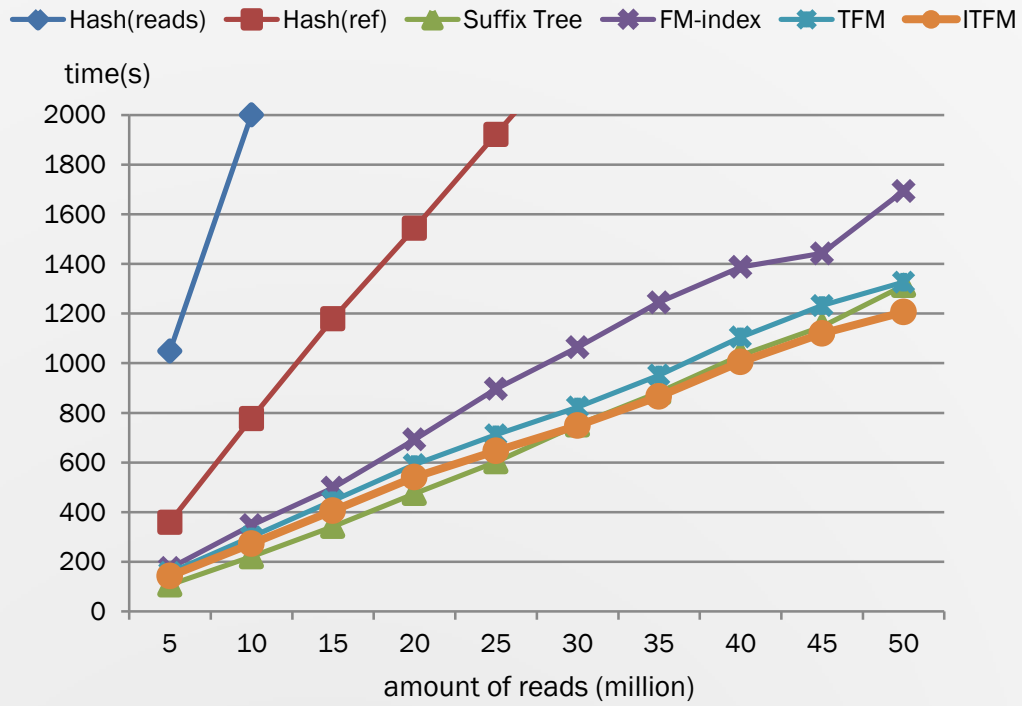
No. of reads (bp)	30M	35M	40M	45M	50M
Time for trie construction	61s	73s	82s	95s	110s

No. of reads (bp)	30M	35M	40M	45M	50M
TFM	76608K	88885K	101023K	113035K	124920K
ITFM	72011K	81774K	91425K	101731K	111553K



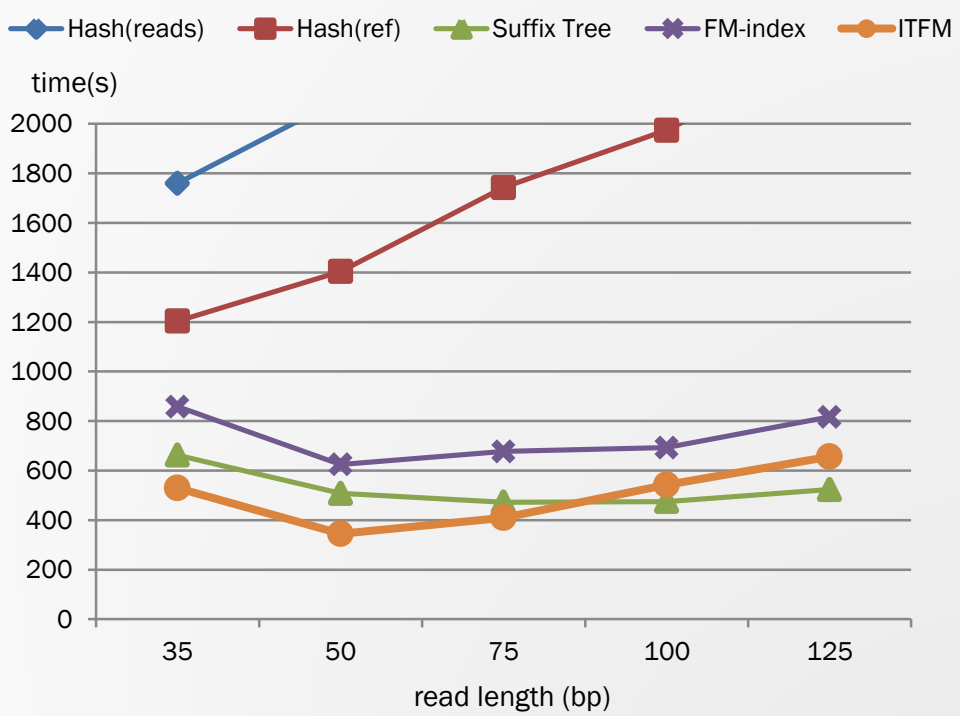
Varying Read Amount

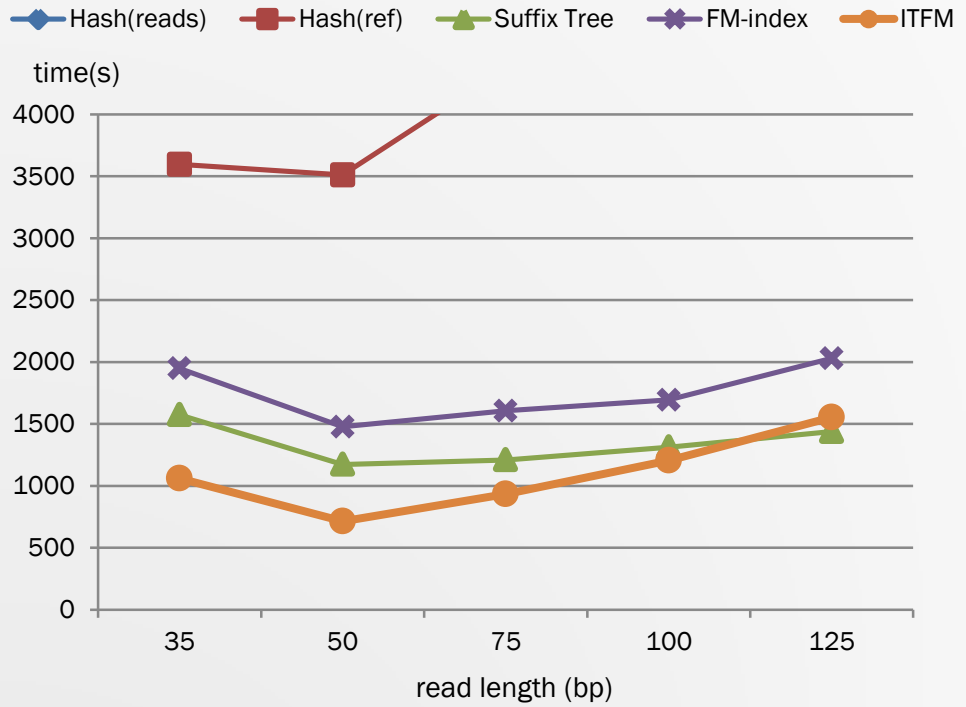
- Genome size = chromosome 1 of Rat genome, 290,094,217 bp.
- Read length = 100 bp



Varying Read Length

- Genome size = chromosome 1 of Rat genome, 290,094,217 bp.
- Read amount = 20 and 50 million





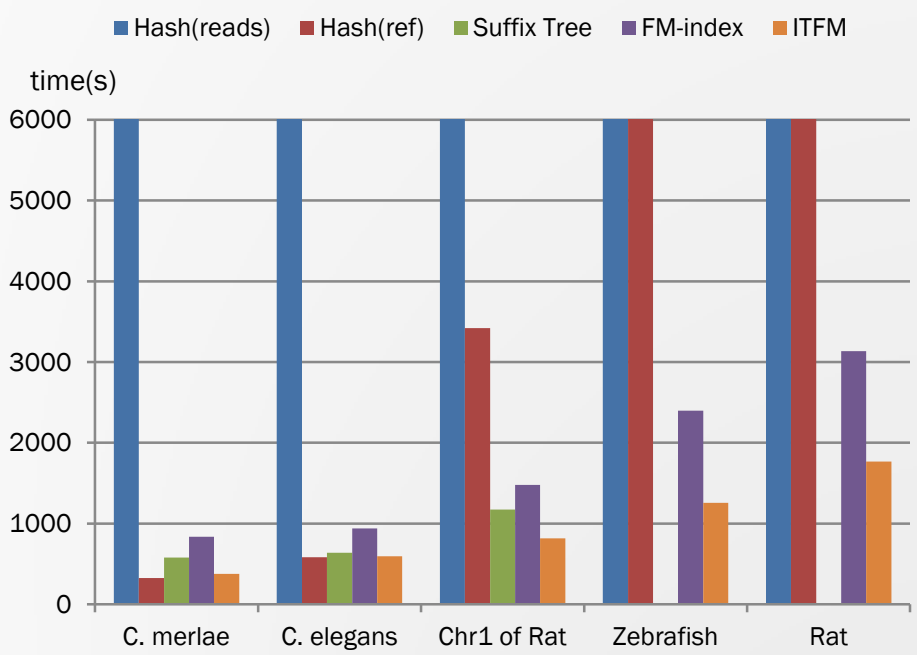
Varying Genome Size

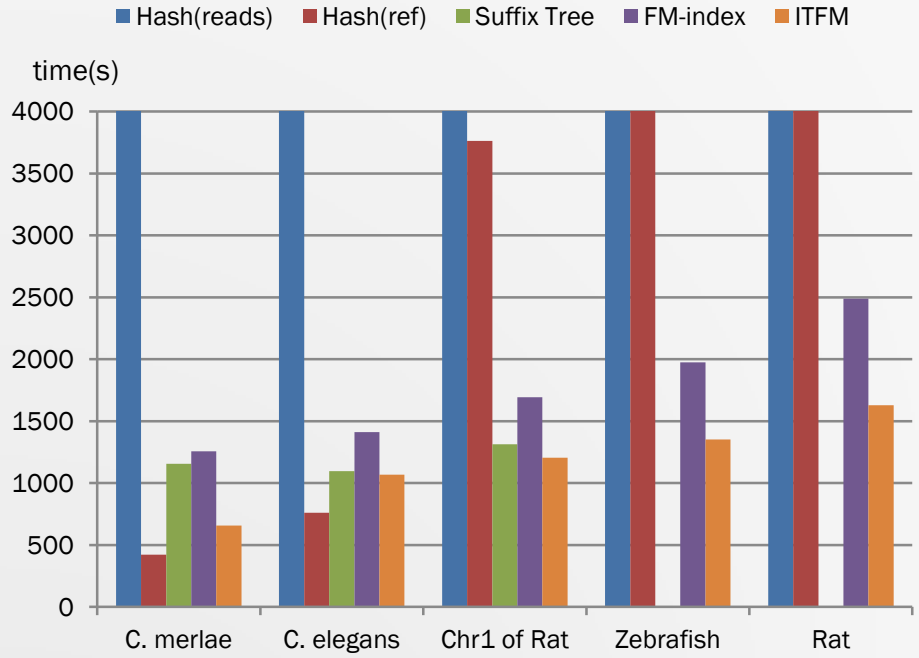
- 5 different genomes:

Genome Name	Genome Size (bp)
C. merlae (ASM9120v1)	16,728,967
C. elegans (WBcel235)	103,022,290
Rat chromosome 1 (Rnor_6.0)	290,094,217
Zerbra fish (GRCz10)	1,464,443,456
Rat (Rnor_6.0)	2,909,701,677

Varying Genome Size

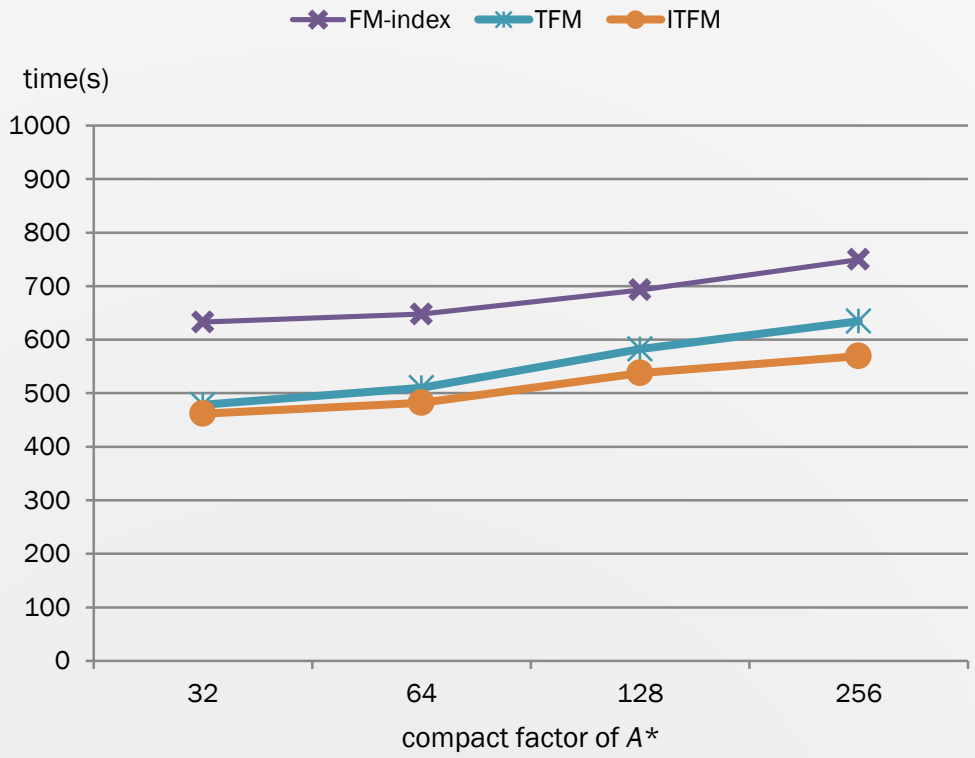
- Read amount = 50 million.
- Read length = 50 bp and 100 bp.





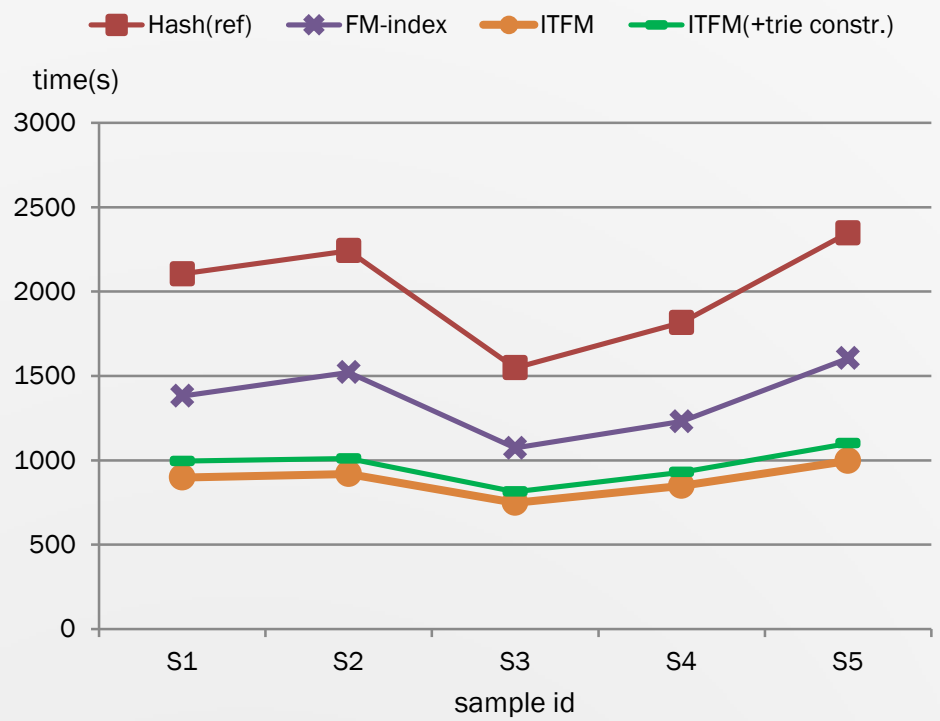
Varying Bucket Size of Appearance Array

- Read amount = 20 million.
- Read length = 100 bp.



Experiments with Real Data

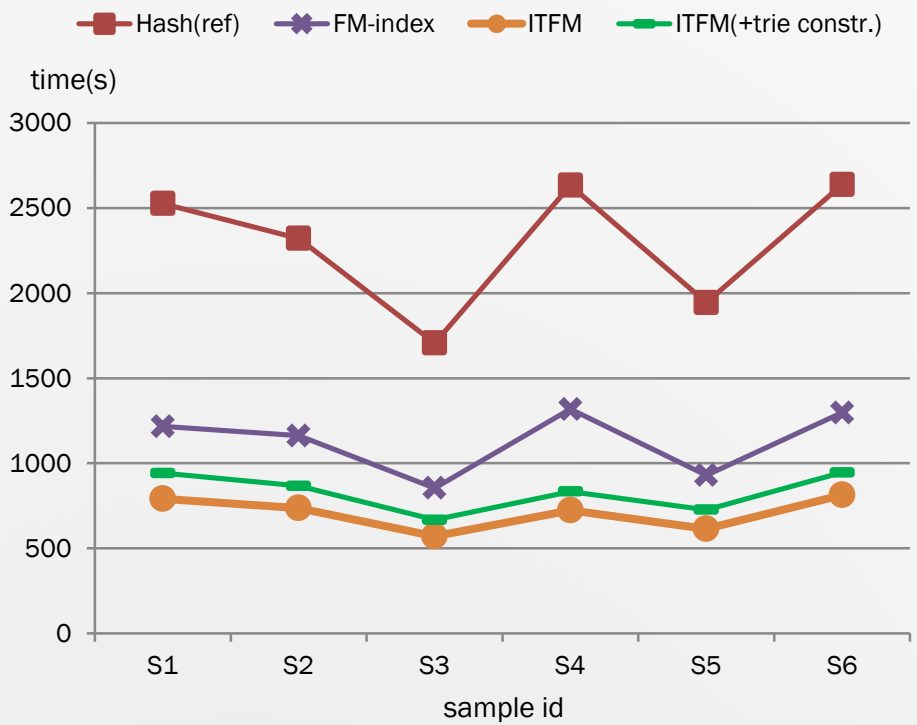
- Dataset: 5 rat samples [10]
- Read length = 50 bp



Sample ID	S1	S2	S3	S4	S5
No. of reads (bp)	63,058,350	70,902,476	46,768,753	52,830,741	73,558,762

Experiments with Real Data

- Dataset: 6 rat samples [10]
- Read length = 36-100 bp



Sample ID	S1	S2	S3	S4	S5	S6
No. of reads	71,160,190	66,203,093	47,937,592	74,941,568	53,839,641	74,663,544

Experiment of Inexact Mapping

- Read length = 50 bp. Read amount = 46,768,753.
- Mismatches allowed = 3.
- Methods Compared:
  - Our method, denoted by *ITFM*.
  - Hash table constructed over reference genome, denoted by *Hash Table (reference)*.
  - *FM*-index start inexact search when exact matching fails, denoted by *FM-index (break point)*.
  - *FM*-index start inexact search from 10^{th} base of reads, denoted by *FM-index (10^{th} base)*.

Experiment Result of Inexact Mapping

Method	Time (s)	Mapping Rate
ITFM	1573	85.3%
FM-index (break point)	1426	84.4%
FM-index (10^{th} base)	2208	85.5%
Hash table (reference)	2320	87%

Memory Usage

- Hash table constructed over Rat genome : ~13 Gb.
- FM-index for Rat genome: ~5 Gb.
- Our method: ~14.2 Gb.
  - FM-index: ~5 Gb.
  - Trie for ~50 million 100 bp reads: around ~9.2 Gb.

Conclusion

- Introduced DNA sequencing technologies.
- Reviewed related short-reads mapping approaches.
- Presented the method combining trie and FM-index for matching massive short-reads.
- Experiment results demonstrated that our method can reduce the running time of the traditional FM-index search for big set of short-reads for mammalian-sized genome databases.

Future Work

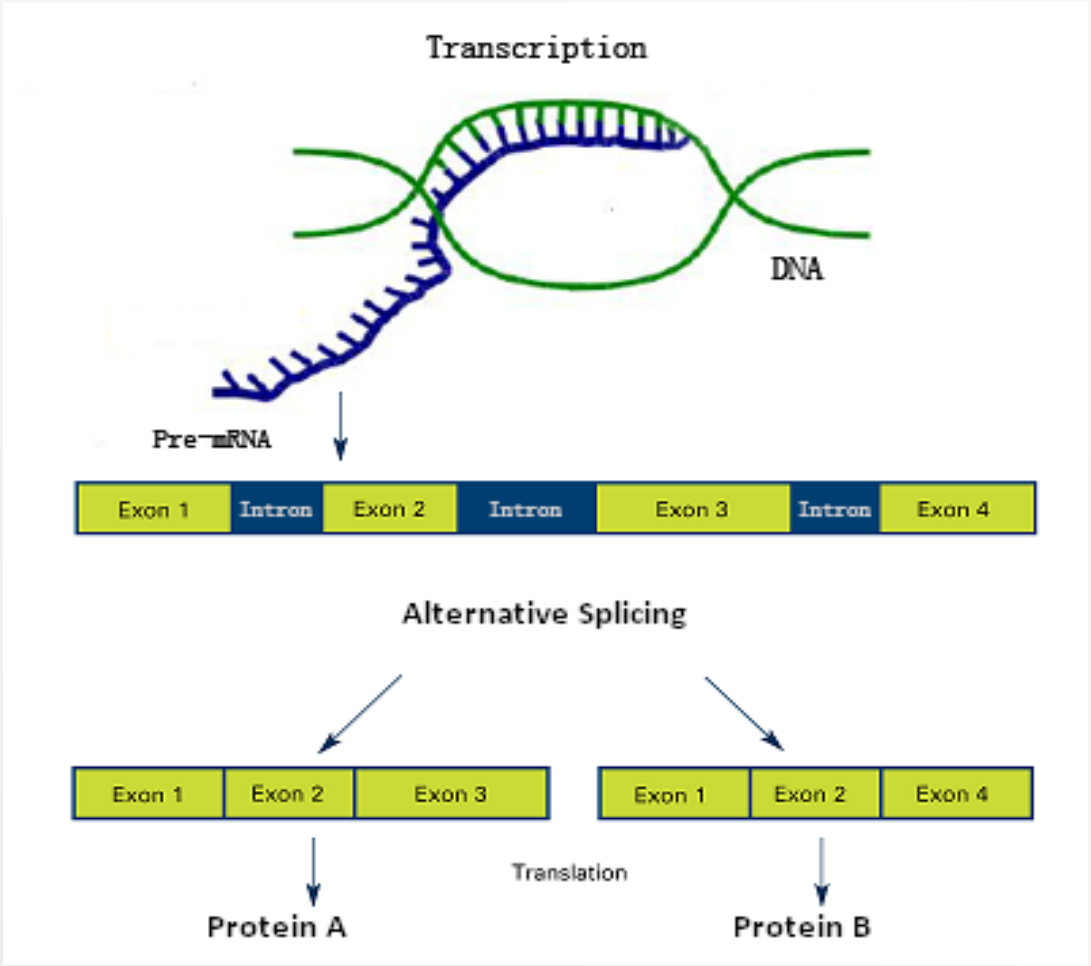
- Further reduce memory usage of trie.
- Adapt our matching algorithm for protein sequences.
- Introducing mapping quality, rank matches by mapping quality.

References

- [1] S. Andrews, “Babraham Bioinformatics - FastQC,” 2010, http://www.bioinformatics.babraham.ac.uk/projects/fastqc/.
- [2] Bolger, A. and Giorgi, F., Trimmomatic: A flexible read trimming tool for Illumina NGS data. URL http://www.usadellab.org/cms/index.php?page=trimmomatic.
- [3] Langmead, Ben et al. “Ultrafast and Memory-Efficient Alignment of Short DNA Sequences to the Human Genome.” *Genome Biology* 10.3 (2009).
- [4] D. Kim, G. Pertea, C. Trapnell, H. Pimentel, R. Kelley, and S. L. Salzberg, “TopHat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions.” Genome biology, vol. 14, no. 4, p. R36, Apr. 2013.
- [5] Trapnell, C., Williams, B. a., Pertea, G., Mortazavi, A., Kwan, G., van Baren, M. J., Salzberg, S. L., Wold, B. J. and Pachter, L., Transcript assembly and quantification by RNASeq reveals unannotated transcripts and isoform switching during cell differentiation. Nature Biotechnology, 28,5(2010), pp, 511– 5.
- [6] Robinson MD, McCarthy DJ and Smyth GK (2010). “edgeR: a Bioconductor package for differential expression analysis of digital gene expression data.” Bioinformatics, 26, pp. -1.
- [7] Anders S, Reyes A and Huber W (2012). “Detecting differential usage of exons from RNA-seq data.” Genome Research, 22, pp. 4025.
- [8] P. Ferragina and G. Manzini, Opportunistic data structures with applications. In Proc. 41st Annual Symposium on Foundations of Computer Science, pp. 390 - 398. IEEE, 2000.
- [9 H. Li, wgsim: a small tool for simulating sequence reads from a reference genome, https://github.com/lh3/wgsim/, 2014.
- [10] Xie’s lab website: http://home.cc.umanitoba.ca/~xiej/, 2014.
- [11] Li H, Ruan J, Durbin R. Mapping short DNA sequencing reads and calling variants using mapping quality scores. Genome research ,2008,18(11): 1851-1858.
- [12] Li R, Li Y, Kristiansen K, Wang J. SOAP: short oligonucleotide alignment program. Bioinformatics, 2008,24(5):713-714.
- [13] S Bauer, M H Schulz, P N Robinson, gsuffix, URL http:://gsuffix.Sourceforge.net/, 2014.
- [14] Wu, Z., Jia, X., de la Cruz, L., Su, X.C., Marzolf, B., Troisch, P., Zak, D., Hamilton, A., Whittle, B., Yu, D., Sheahan, D., Bertram. (2008). Immunity 29, this issue, 863–875.
- [15] J. C. Venter, M. D. Adams, and E. W. Myers et al. The sequence of the human genome. Science, 291(5507):1304–1351, Feb 2001.

Biology Background

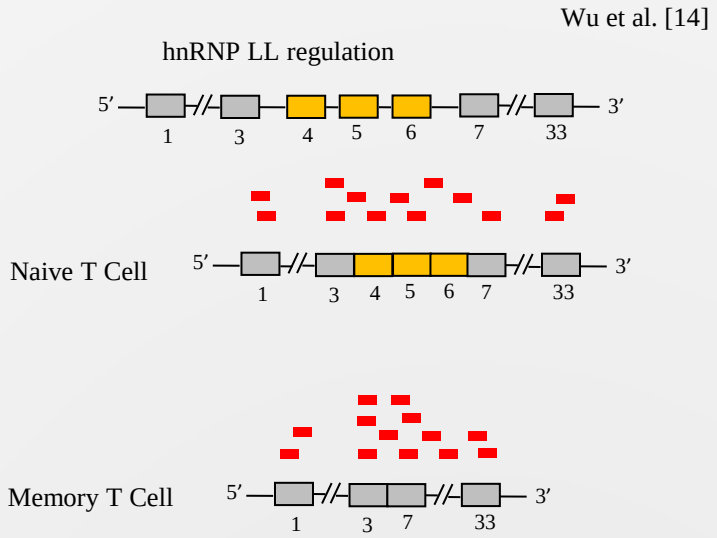
- DNA
- Gene
  - Exon
  - Intron
- Alternative splicing
- Transcript



[15]

Differential Alternative Splicing Analysis

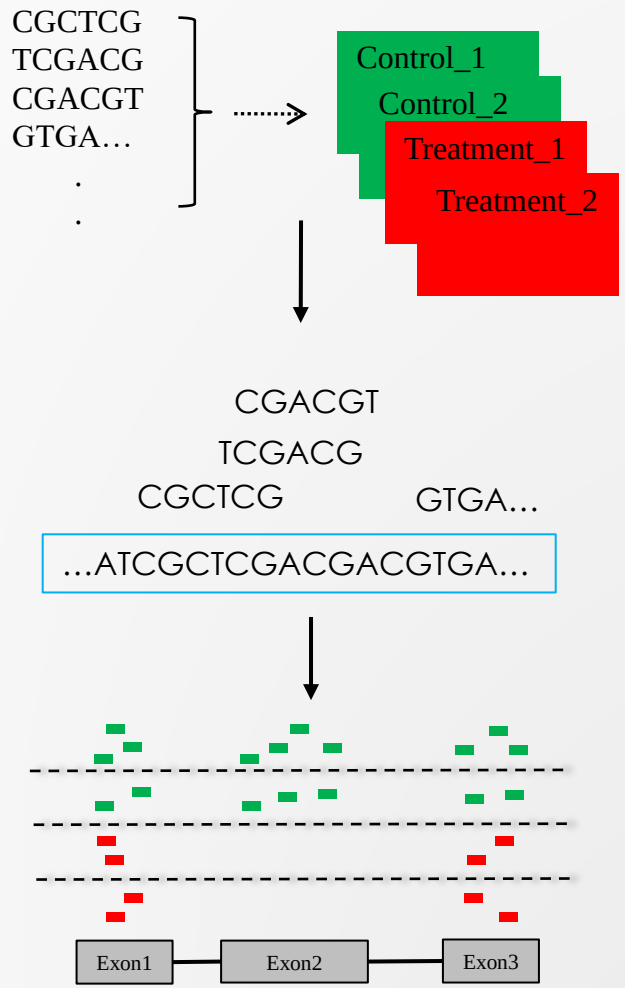
- Find differences in exon splicing patterns among different biological conditions.
- Detect the differences by analyzing distribution of short-reads (expression level).



Pipeline Tool Motivation

- Analyzing NGS data is complicated.
  - Multiple phases are needed.
  - Differential alternative splicing analysis is not settled down into definite “best practice”, several methods are available.
  - Typically many samples in an experiment will be processed in the same way.

Pipeline Workflow



Raw reads

Cleaning reads

- •Quality assessment for whole library.
- •Trim low quality 3' end sequences.
- •Remove low quality reads.
- •Remove overrepresented sequences.

Mapping reads

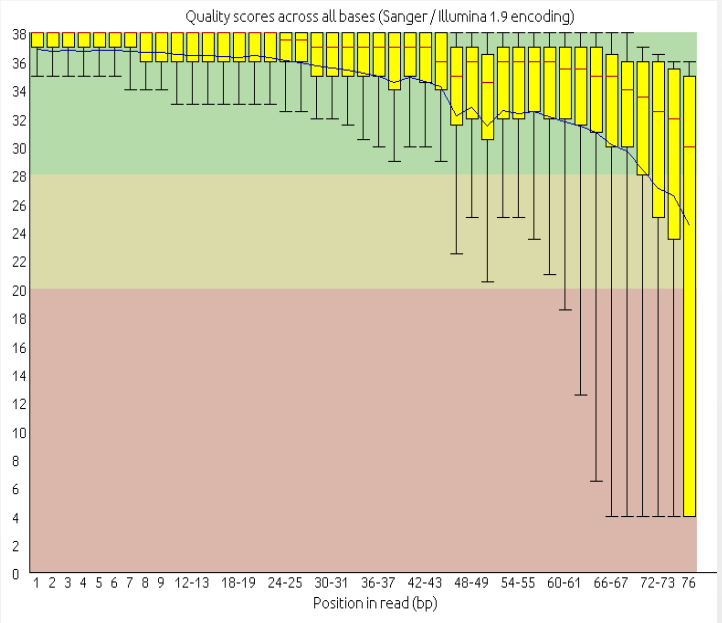
- •Only reads that pass the cleaning step are remained.
- •Map cleaned short-reads against transcriptome.

Expression analysis

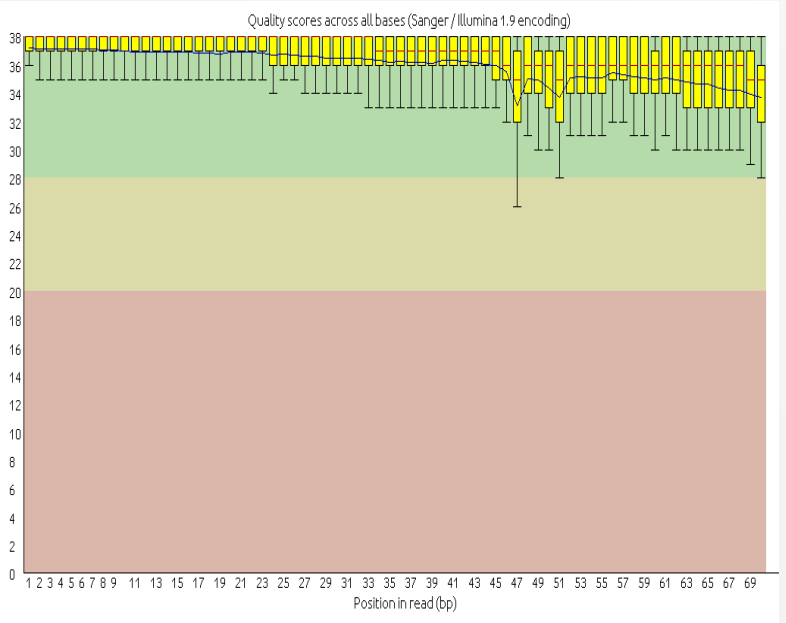
- •Count mapped reads for each element at exon level.
- •Detect alternative splicing by exact test.
- •Filter counting bins with low counts
- •Report a table of detected alternative splice sites.

Cleaning Raw Reads

- Sequencer may generate poor quality reads.
  - Use FastQC [1] to assess quality of reads.
  - Use Trimmomatic [2] to clean reads: trailing quality < 28, minimum length = 32 bp.



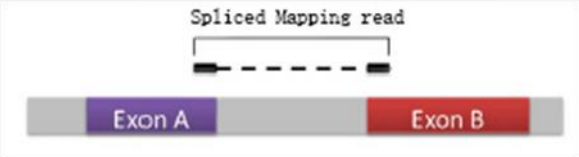
before cleaning



after cleaning

Strategies of mapping

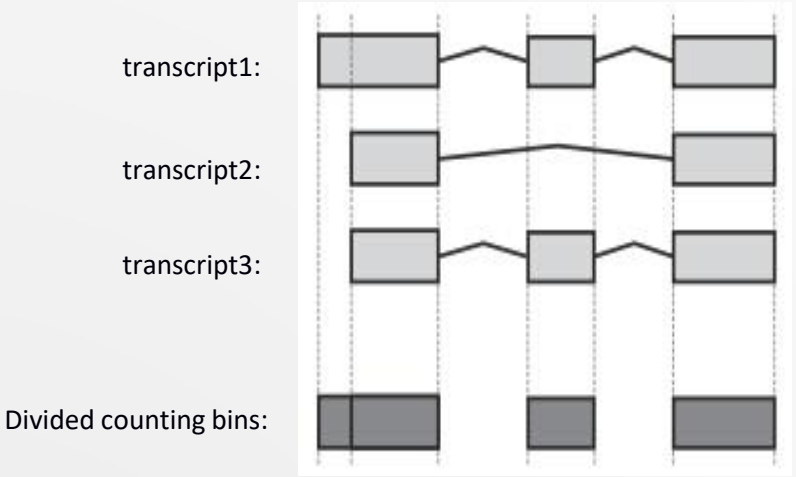
- Unspliced mapper.
  - Bowtie [3]. Best for analysis within known genes.
- Spliced mapper.
  - Tophat [4]. Best for unknown exon, gene detection.



- We use Bowtie in our pipeline.
  - Map reads to transcriptome.
  - Increase accurate rate of mapping.
  - Increase mapping speed.

Strategies of Quantitative Evaluation

- Transcript estimation based. e.g. Cufflinks [5].
  - Direct way.
  - Lack of accuracy.
- Count at gene level. e.g. edgeR [6].
  - Simple.
  - Miss many results.
- Count at exon level. e.g. DEXSeq [7]



Filter Results

- Counting bins with low number of reads assigned may be wrongly detected.
- We shouldn't merely rule out counting bins by raw counts, as the influence of sequencing depth should be considered.
- Use normalization model *count per million* (*CPM*).
  - $CPM_e = r_e \times (10^6 / N)$
- Filter out counting bins with $\overline{CPM} < 0.2$ in all experimental groups.

Comparison with Existing Pipeline

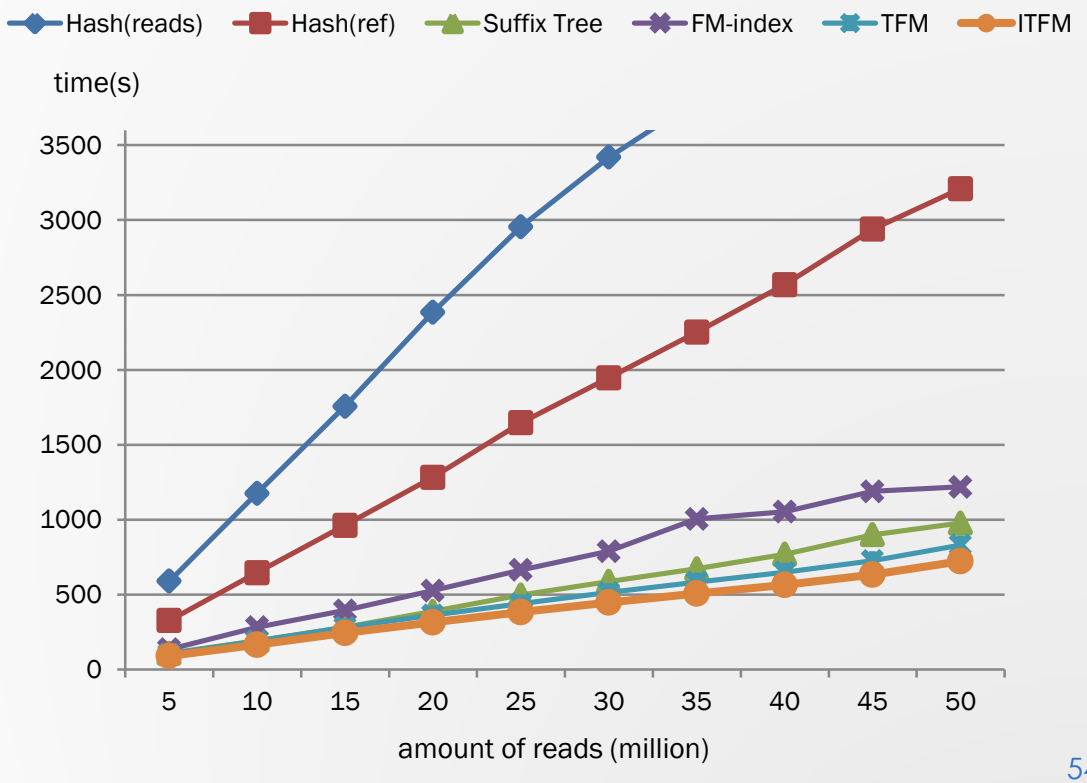
- Implementation: Combine python, bash, R scripts incorporating publicly available tools.
- Compare with Tophat-Cufflinks pipeline. Dataset: hnRNP L&LL regulation[10]

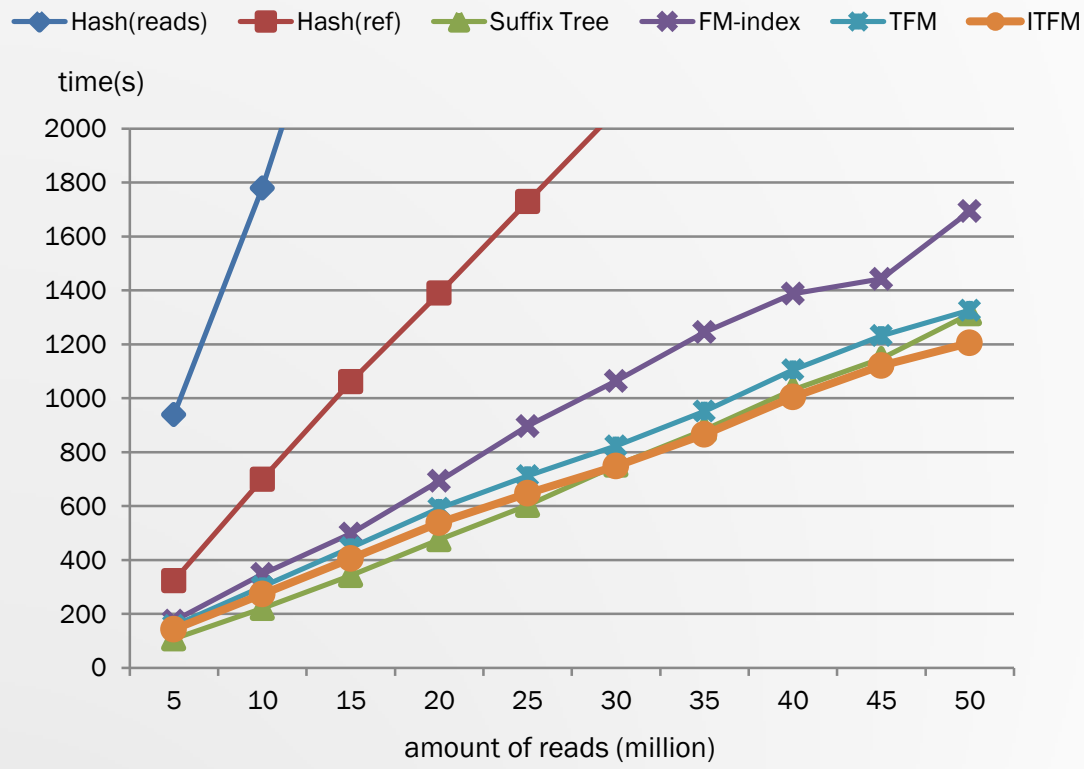
Analytical category	Pipeline name	Performance	Time
Preprocessing	Ours (FastQC)	85% good quality reads	47 min
Preprocessing	T-C (FastQC)	85% good quality reads	47 min
Read mapping	Ours (Bowtie)	86.5% reads mapped	7 h
Read mapping	T-C (Tophat)	93.2% reads mapped	13 h
Differential expression	Ours (DEXSeq)	270 bins differentially used	2 h
Differential expression	T-C (Cufflinks)	607 transcripts differentially expressed	5 h

Experimental Validation:
Ours: 60% (6 out of 10)
T-C: 30% (3 out of 10)

Varying Read Amount

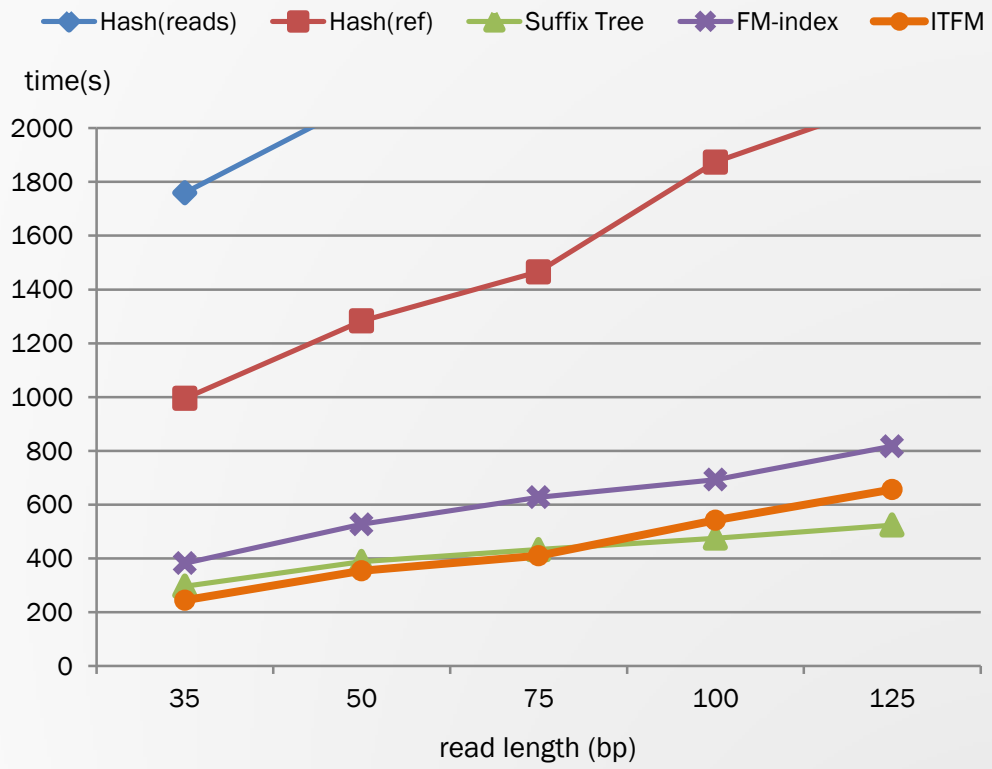
- Genome size = chromosome 1 of Rat genome, 290,094,217 bp.
- Read length = 50 and100 bp
- Find 10 appearance locations

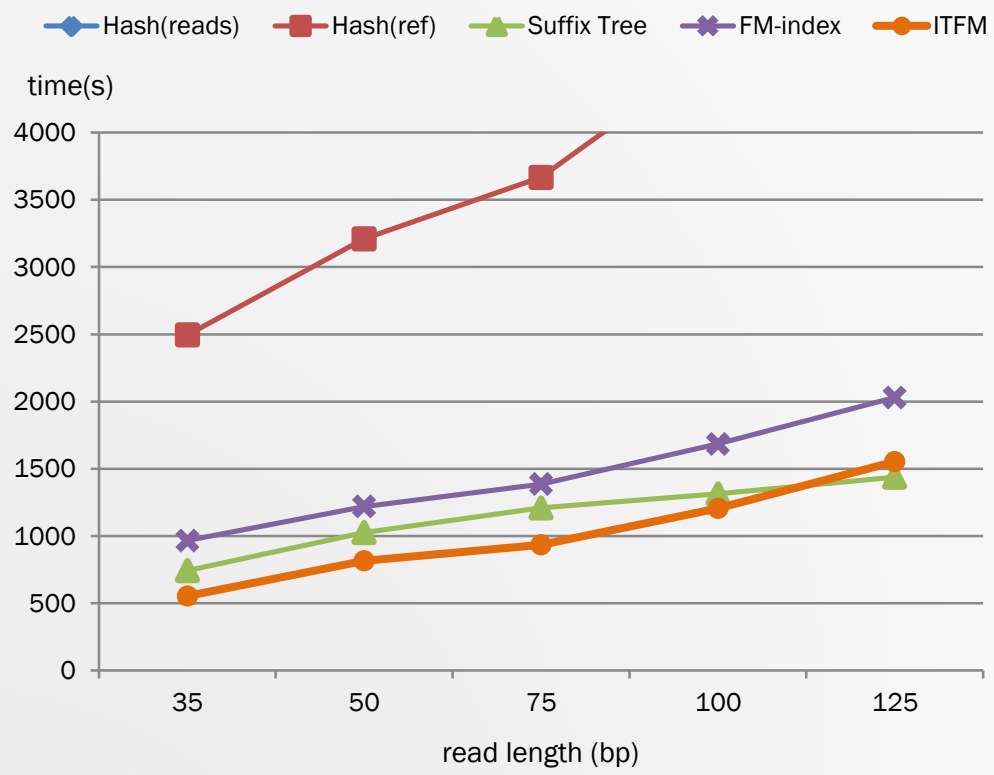




Varying Read Length

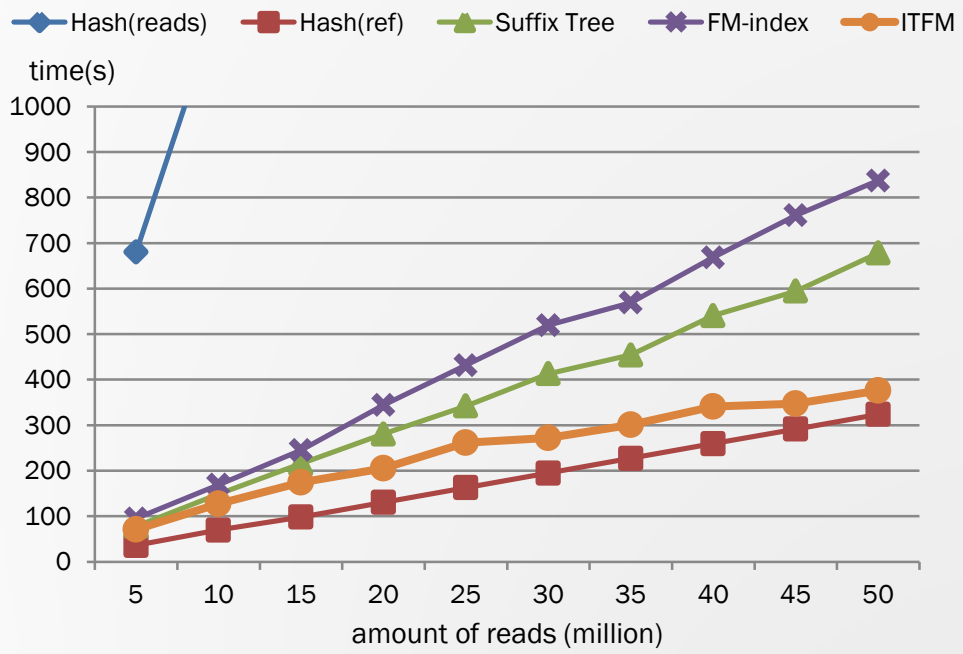
- Genome size = chromosome 1 of Rat genome, 290,094,217 bp.
- Read amount = 20 and 50 million
- Find 10 appearance locations

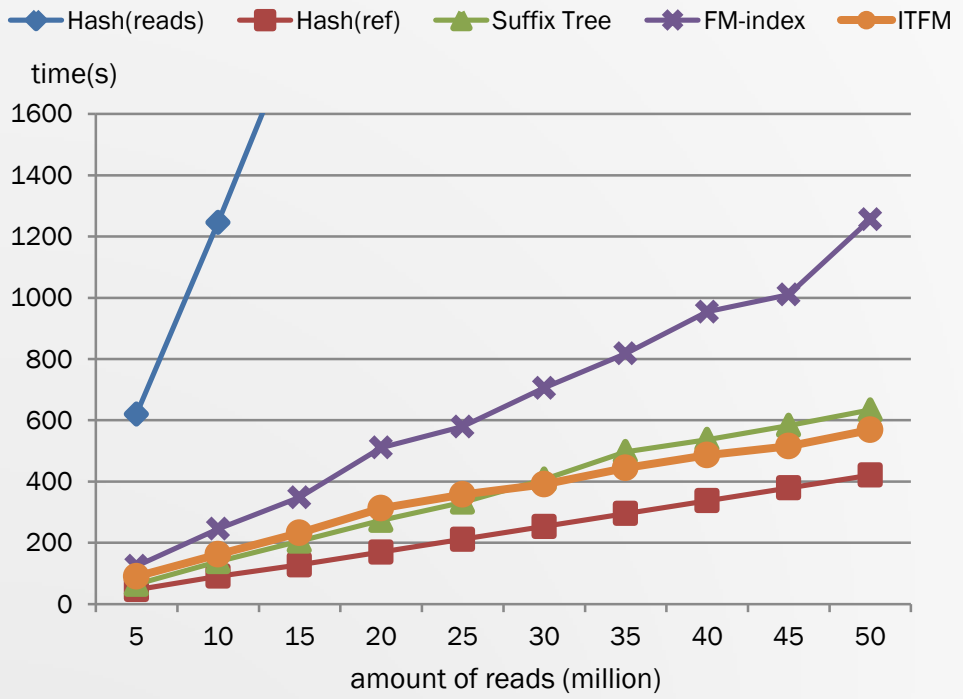




Varying Read Amount

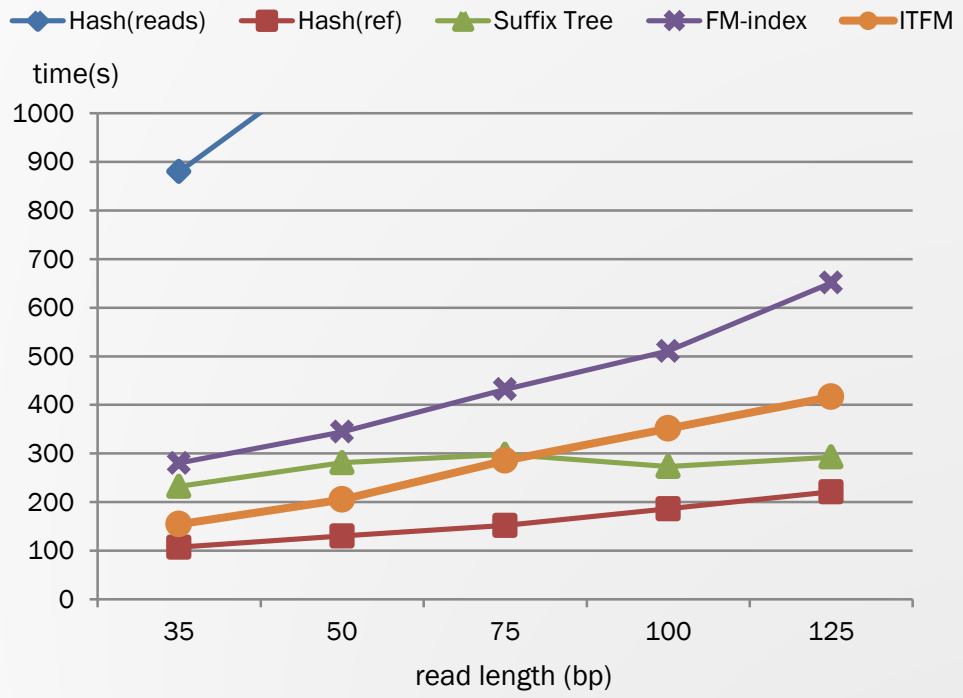
- Genome size = C. merlae, 16,728,967 bp.
- Read length = 50 and100 bp

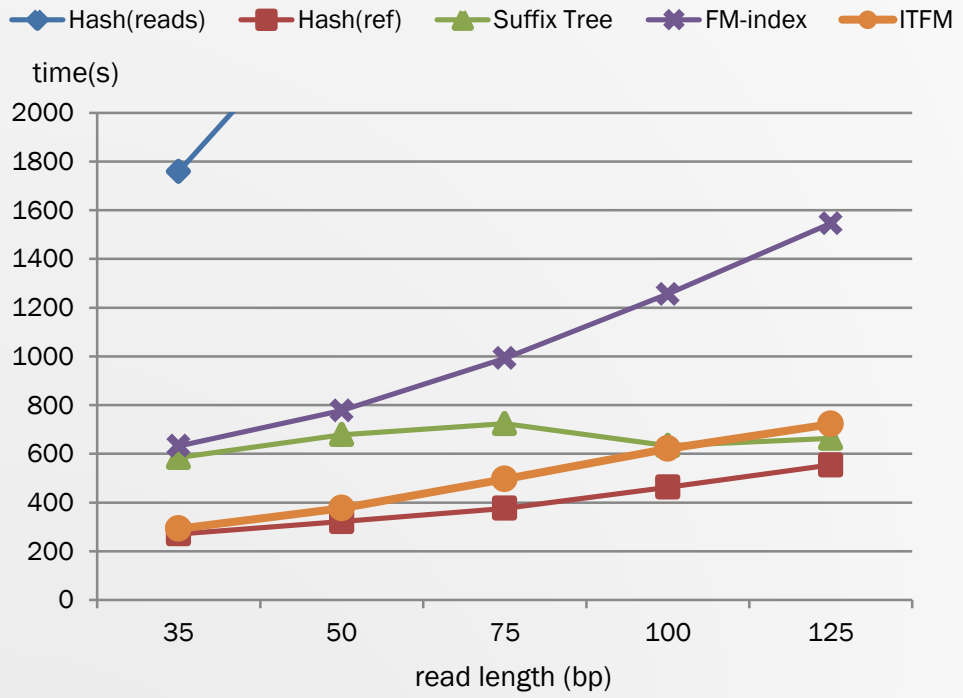




Varying Read Length

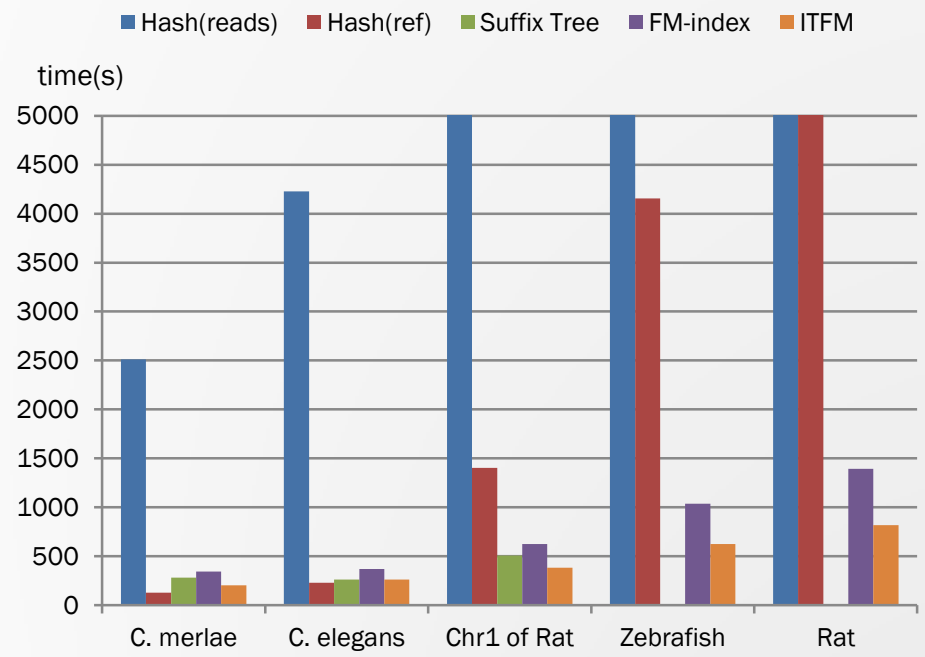
- Genome size = C. merlae, 16,728,967 bp.
- Read amount = 20 and 50 million

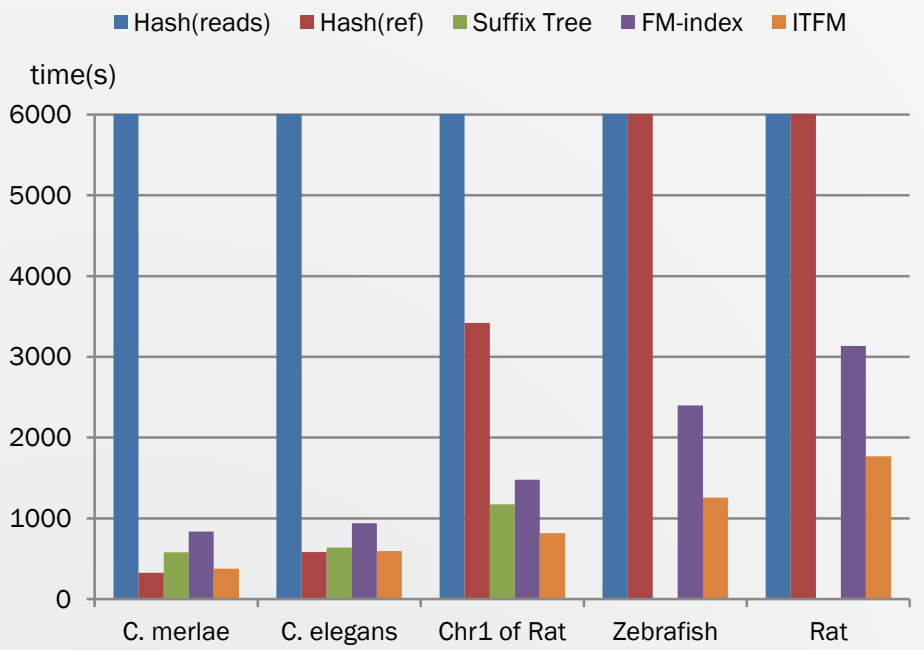




Varying Genome Size

- Read amount = 20 and 50 million.
- Read length = 50 bp.





Varying Genome Size

- Read amount = 20 and 50 million.
- Read length =100 bp.