UNIVERSITY OF ZAGREB

FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

MASTER THESIS ASSIGNMENT No. 744

RNA-seq mapper

Igor Jerković

Zagreb, June 2014

Special thanks to Mile Šikić for valuable advices and being my mentor

Table of contents

1. Introduction

This thesis describes a tool for mapping RNA-seq data that was developed as a part of the assignment. It is divided into 4 main parts:

- First, a short introduction of the problem and its bioinformatics background is given
- In the second part the algorithms used in the tool are being described
- After that, results and comparison with other well-known tools and solutions is being presented
- At the end, there is a short conclusion and notes for future work

1.1. RNA-seq

RNA sequencing is a method for transcriptome analysis that uses capabilites of next-generation sequencing that captures a snapshot of RNA from the genome. Complementary DNA (cDNA) generated from RNA are sequenced using next-generation technologies, that are usually 'short read' sequencers [1][2]. This is important because the problem is in a way similair to the problem of mapping DNA reads to the genome. We can model the RNA as a linear array of bases. Because of cDNA is used there are exactly four different bases that are marked *A*, *C*, *G* and *T* (short for adenine, cytosine, guanine and thymine). The additional problem to mapping the reads to the genome that DNA sequencing has, in RNA sequencing is the problem of spliced reads. RNA reads are being obtained from the processed mRNA after the introns are being cut off. Spliced reads are those that span over at least two exons. The additional problem can now be seen much easier because not only reads have to be correctly mapped to the genome, reads that are cut at some places need to be mapped correctly as well.

An exon is a region of DNA within a gene that is transcribed to the final messenger RNA (mRNA) molecule. Introns are sections of DNA colinear to the RNA sequence that will be spliced out after transcription, but before the RNA is used. Introns are common in eukaryotic RNAs. The regions of a gene that remain in spliced mRNA are called exons [3]. The number and length of introns varies among species and among genes of the same species. This information can be used when mapping RNA reads against the genome. As well, there are known intron donor-acceptor sites that can be also used into detecting introns and consequently exons as well.

With all this described, it becomes clear that the main problem in mapping the RNA reads against the genome is detection of exon regions from which the reads were obtained. Sites where exons join are called junctions.

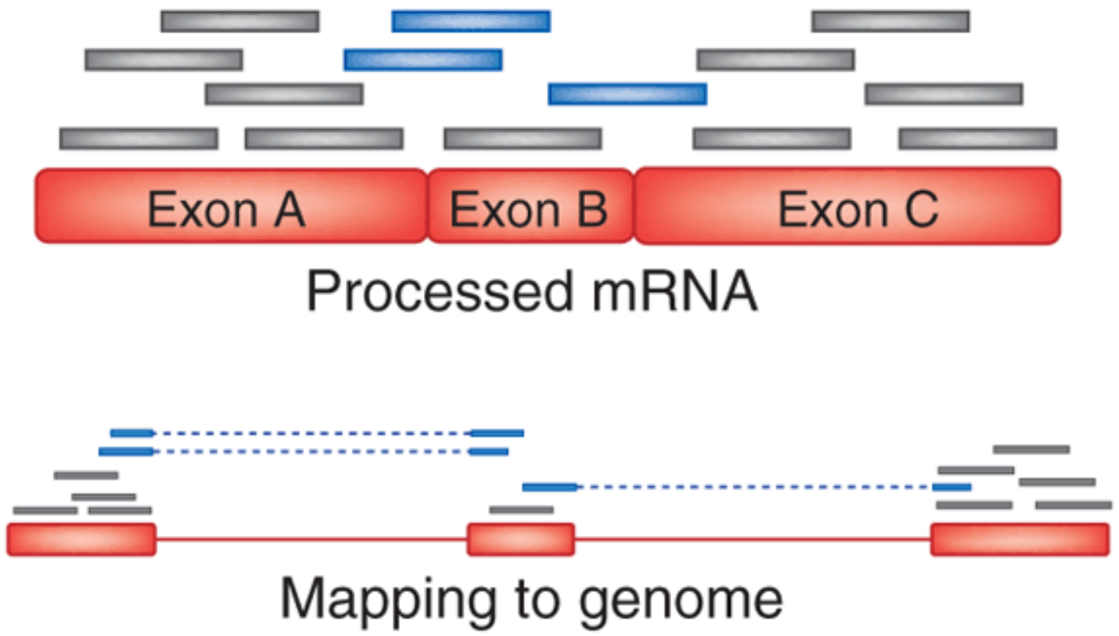

Figure 1. reads obtained from processed mRNA

On *Figure 1* the explained problem is shown. Gray reads were obtained from one exon region, while reads colored in blue are spanning two exons. The main problem is the detection of mapping position of the spanning reads. Solution used in this thesis is described in the next section were the algorithms used behind the tool are being explained.

Except the main difference between DNA and RNA mapping that is spliced mapping, old problems for DNA mapping remain. Such as insertions, deletions, supstitutions, inversion and doubling the parts of reads.

The final goal of RNA sequencing is to align the reads against the reference genome and to construct a transcriptome map. The transcriptome is the complete set of transcripts in a cell. Understanding the transcriptome is essential for various reasons such as: interpreting the functional elements of the genome, revealing the molecular constituents of cell and tissues, undrestanding development of diseases etc.

2. Methods

2.1. Overview of current RNA-seq mappers

In this section an overview of the best current RNA-seq tools is given. After that, approach used in the tool that was developed as part of the asssignment and algorithms behind it are being described. When it comes down to RNA-seq mappers, there are two main types of them, short (unspliced) aligners and spliced aligners. On this thesis much bigger emphasis is given on the spliced aligners, but first a brief description of unspliced aligners follows.

2.2. Unspliced aligners

Unspliced aligners are often called short aligners. This is beacuse they are able to align only continous reads to a referent genome that does not contain gaps that were results of splicing. Most of those aligners are one of two types:

1. Based on the Burrows-Wheeler transform (such as Bowtie or BWA)
2. Based on seed-extend methods, Needleman-Wunsch or Smith-Waterman algorithms

The first group is many times faster (especially Bowtie, Bowtie2 or BWA), but some tools of the second group tend to be more sensitive. However, since in this thesis the assignment was to build a spliced aligner, more about them follows.

2.3. Spliced aligners

Many reads can not be aligned directly by unspliced (short) aligners because many reads span exon-exon junctions. This is where spliced aligners are needed. Some of the aligners of this group use unspliced aligners to firstly align continuous reads and to detect exons. After that they use a different strategy to match rest of the reads that span over spliced regions. Mostly, those reads are split into smaller segments and mapped independently afterwards. Also, among the spliced aligners two types of aligners exist: annotation-guided aligners (aligners based on known splice junctions) and *De novo* splice aligners. For annotation-guided aligners the detection of splice junctions is based on data available in databases about known junctions. This type of aligners can not detect new splice junctions. Most famous aligners of this type are RUM and SpliceSeq. With RUM, reads are first mapped with Bowtie against the genome and transcriptome, also an annotation file is needed to be provided in the pipeline this tool runs. It also uses BLAT mappings that are afterwards merged with Bowtie mappings for the final alignments. At the end coverage and junction files are produced.

However, in this thesis main focus is on *De novo* aligners that do not need previously annotated information, as a tool that does not need annotated files was developed. Probably the best aligners that fall into this category are Tophat, BBMap, MapSplice, Subread (and Subjunc), GSNAP and STAR.

Tophat alignes reads in two steps. In the first step unspliced reads are aligned with Bowtie. After that reads are assembled with Maq resulting islands of sequences. Those islands are considered to be candidates for exon regions. In the second step, splice junction sites are being determined based on the initially unmapped reads. Biological background of introns is being used in this step, such as well known canonical donor and acceptor sites within the island sequences or distribution of length of introns and exons. There is also a step that could be called step zero. If the annotation file is provided Tophat uses that information to improve overall sensitivity and accuracy of mapping. It also gives the whole pipeline a significant speed increase, as smaller amount of reads need to be mapped in next steps [4][5].

BBMap uses short kmers to align reads directly to the genome. It is highly tolerant of errors caused by substitutions and indels. It claims to be faster and more sensitive than Tophat. It does not use any splice site finding heuristics optimized for a specific taxonomic branch. It finds optimally-scoring affine-transform global alignments, and this makes it ideal for studying new organisms that have no annotation information provided before [6].

Mapsplice is good in detection of small exons, as well as discovery of canonical, semi-canonical and non-canonical junctions. It has two main phases called tag alignment and splice inference phase. In the first phase candidate alignments of the mRNA tags to the reference genome are determined. In the second phase splice junctions that appear in the alignments of one or more tags are analyzed to determine a splice significance score. At the end files in SAM format describing alignments are produced like most other aligners do [7].

Subread uses a slightly different approach. It first breaks down the reads into smaller chunks, thus the name subread. From the relatively large number of short seeds extracted from each read, a strategy called seed-and-vote is being applied. The strategy is sensitive because individual subreads are not required to match exaclyl, rather sites that are considered to be map positions for particular reads need to be supported by several subreads by the vote phase. This can get easily extended to finding exon-exon junctions, by locating reads that contain sets of subreads mapping to different exon of the same gene. Subread also uses donor and acceptor sites for considering splice sites [8].

GSNAP has a similar approach, it first builds an index of the genome using a hash table. Afterwards, reads are broken into shorter elements that can be looked up in the hash table. Resulting position lists for each element are being checked to see if they support a common target location and have a reasonable number of mismatches. The number of mismatches is verified by checking the whole read against the reference [9].

STAR is a spliced aligner that uses 'sequential maximum mappable seed search in uncompressed suffix arrays followed by seed clustering and stitching procedure'. It detects both canonical and non-canonical splice junctions. It can align long reads obtained by third-generation sequencing technologies and can reach speed of 45 million paired reads per hour per processor [10].

2.4. Approach

Approach used in developing a new RNA-seq mapper in this thesis can be broken into three main parts:

1. Genome alignment
2. Detecting potential exon regions
3. Spliced alignment

2.4.1. Genome alignment

The main inspiration in this phase is drawn from the fact that reads that were sequenced from a single exon are not spliced and can be matched against the genome with usual alignment methods. Input for this phase is the referent genome and all the reads that need to be aligned, both spliced and unspliced reads. At this point reads that span splice sites are unknown. The premise is that spliced reads won't map well in this phase and are going to be left out to map in next steps when better assumptions about potential exon regions are made. In this phase an index for the genome needs to be built if it is not given beforehand. This step can be done in preprocessing and is not getting measured into the effeciency of the tool and its algorithms overall. In this phase the developed tool can use three aligners:

- Bowtie (Bowtie2)
- BWA
- LISA

By making the option for using few aligners in genome alignment phase wider window for testing has opened as all of those aligners have some specifics. This phase generates initial alignments that are passed to the next phase of detecting potential exon regions. Unmapped reads are going to be processed for the third phase after detection of possible exon regions is done.

2.4.2. Exon regions detection

The second phase is computing of potential exon regions that can be used in the next step for matching initially unmapped reads. Genome alignment step determined which reads are well mapping to the reference genome. Those reads are usually continous reads that did not have bigger errors caused by insertions or deletions though aligners from the first phase do permit some smaller errors. This fact can be used in determining the exon regions as reads that were sequenced from only one exon are going to map well and are going to be supported by other reads that mapped to just that exon as well. A matric called coverage is used in this step for better determination of exon regions. Coverage in RNA sequencing refers to the number of times a nucleotide is read during the sequencing process. Coverage is the average number of reads representing a given nucleotide in the reconstructed sequence. Coverage can be easily calculated from the length of the potential exon region, the number of reads and the average read length.

$$C = N \times \frac{L}{E}$$

Figure 2.

C - coverage,

N - number of the reads mapped on that exon region,

L - average read length,

E - length of the exon region

For example, an exon with 150 base pairs reconstructed from 32 reads with an average length of 100 nucleotides will have 21.33 redundancy (or coverage). A high coverage is often desired when sequencing because it can overcome errors in base calling and assembly. The term deep coverage has been used for various values (usually >10x). There is also a newer term ultra deep coverage that appeared in scientific literature to refer even higher coverage (>100x) [11].

Tophat uses a value of 20 for coverage search in determining the exon regions, that value is also used in this tool, but can be easily adjusted as a passed in option for making experiments and custom evaluations.

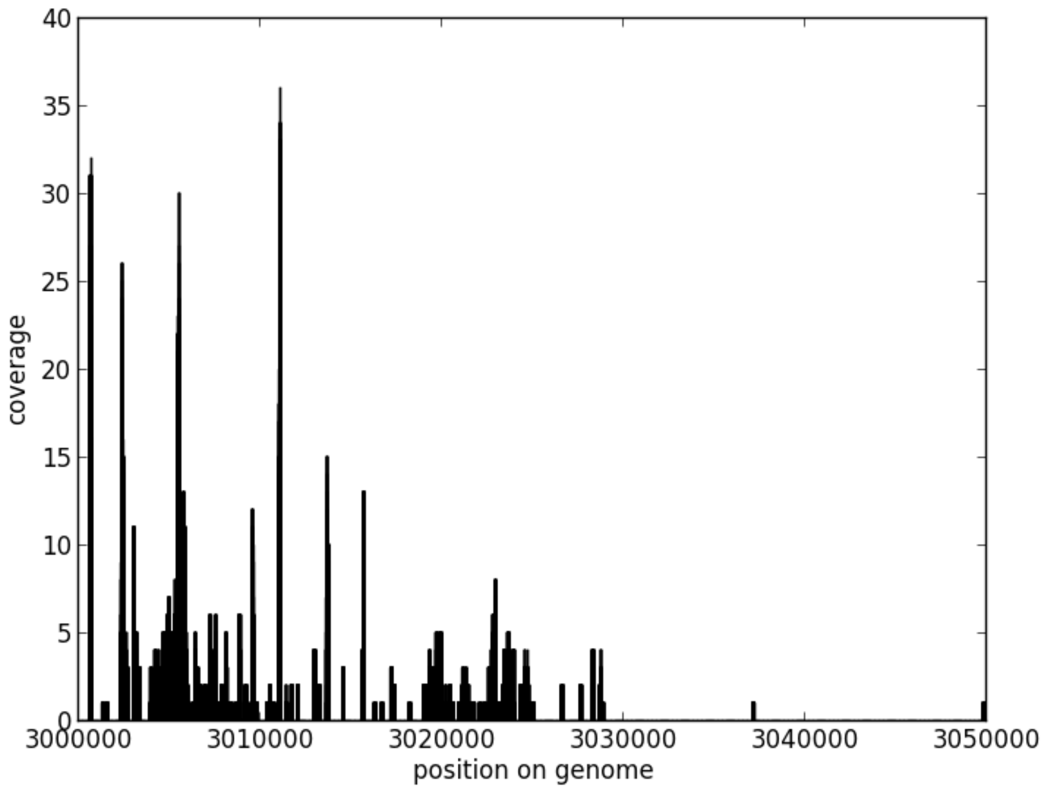

Figure 3. coverage values used to determine exon islands

Figure 3 shows values for coverage that have been drawn from an experiment ran with the new rna-seq mapper. Reads of brain tissue mapping to chromosome 19 of human genome from position 3000000 to 3050000 are shown.

The formula used in calculating coverage in rna-seq mapper for getting possible exon regions:

$$C = \frac{1}{E} \sum_{i=0}^{E} c_i$$

Figure 4.

C – coverage of exon region,

E – length of exon region,

C_i - value of coverage at position i

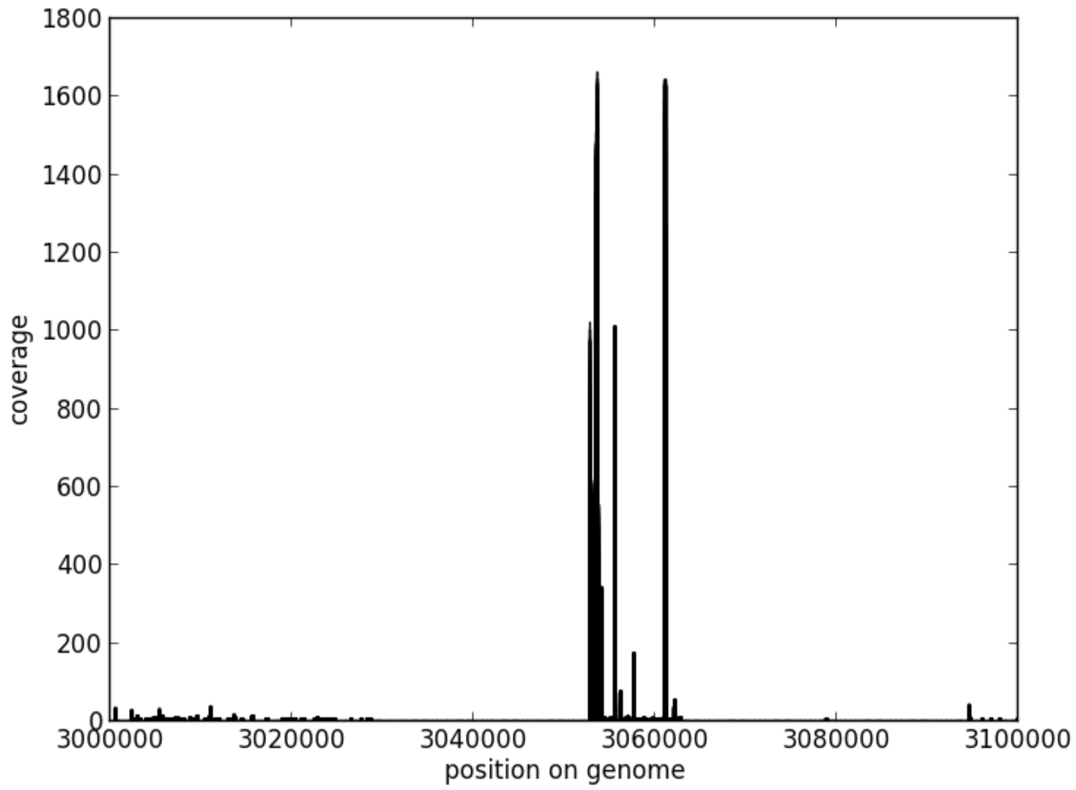

Figure 5. coverage with larger values

Figure 5 also shows coverage values from experiment ran with rna-seq mapper, mapping brain tissue reads to human genome chromosome 19. Around position 3060000, large values of coverage can be seen. This indicates that some regions are more deeply sequenced, but from experimenting with the tool it showed that coverage limit of 20 does decent work in search for exon regions.

Other than coverage, in this step biological background of introns is being used as well. Splice sites can be predicted by well known acceptor/donor sites [12][13]. The most frequent pair of donor and acceptor is 'GT' with 'AG'. This infomation is being used when making boundaries of exon regions, by finding potential pairs of donors and acceptors that are passed to the next phase of the pipeline where the potential splice sites are being examined. To test the assumption of donor/acceptor signals, a script that ran through the introns on chromosome 19 of human genome was executed yielding results shown in the table.

Donor-acceptor	**frequency**	**overall**
GT-AG	65733	60.8521%
CT-AC	40932	37.8926%
CT-GC	375	0.347155%
GC-AG	336	0.311051%
TC-CT	122	0.112941%
TG-GA	81	0.0749854%
GT-AT	76	0.0703567%
AT-AC	67	0.062025%
GT-GC	33	0.0305496%
GT-CA	32	0.0296239%
AA-CC	32	0.0296239%
CC-CC	30	0.0277724%
CC-CT	26	0.0240694%
GT-CT	12	0.011109%
GA-AG	12	0.011109%
GA-GA	11	0.0101832%
GC-AA	9	0.00833171%
CT-CC	9	0.00833171%
AG-CC	9	0.00833171%
GT-TA	8	0.00740597%
GT-AC	8	0.00740597%
GG-CC	8	0.00740597%
CT-TC	7	0.00648022%
TT-AG	6	0.00555448%
CT-AT	6	0.00555448%
CA-CA	5	0.00462873%
CA-AC	5	0.00462873%
GG-AG	4	0.00370298%
CC-AG	4	0.00370298%
CA-TC	4	0.00370298%
AT-AG	4	0.00370298%
CC-AC	3	0.00277724%
GT-GT	2	0.00185149%
CT-TT	2	0.00185149%
TT-TC	1	0.000925746%
TT-AT	1	0.000925746%
TG-CT	1	0.000925746%
GT-CC	1	0.000925746%
GG-AC	1	0.000925746%
CT-GG	1	0.000925746%
CT-AA	1	0.000925746%
AC-GC	1	0.000925746%

Figure 6. Donor-acceptor statistics

Figure 6 shows the frequency of donor-acceptor pairs on chromosome 19 of human genome. Out of possible 108021 splice sites for this chromosome almost 98.75% (98.7447%) are two donor-acceptor sites: the most frequent 'AG'-'GT' (60.85%) and 'CT'-'AC' (37.89%). That information is being used as a heuristic in the search for potetential splice sites.

Except the donor-acceptor sites, there is a well-known distribution of lengths of introns on the human genome. That information can be used as well, and passed in as an option to the tool. For example, if two exons are detected to be too close they can be merged. This threshold can be set manually. Also there is a threshold that can be given for widening the search near potential exon regions, so that custom experiments can be ran.

Finally, after all the computation with finding potential exon regions is done, that data is passed for the final phase of the rna-seq mapper, that is called spliced alignment. At this point potential exon regions are passed for further examination, reads mapped with short aligners were part of building those potential exon islands. All the other initially unmapped reads are passed to this phase as well for further alignment.

2.4.3. Spliced alignment

In this phase initially unmapped reads are going to be mapped against the exon regions computed in the previous step. The main problem here is that it is not certain that the exon-exon borders were computed correctly in the previous step, as some heuristics were used, such as donor-acceptor sites, merging exons that were close or setting left and right thresholds for making the search around exons wider.

In order to align the initially unmapped reads finding the longest increasing subsequence is being used. The main motivation for this approach comes from few interesting characteristics of the problem. The reads can have some small error as part of how they were sequenced, but the interesting thing here is that there will be many continous sequences that do not have any error at all. So it turns out that the measure for quality of the mapped reads can be the longest increasing subsequence. The basics of the idea are taken from [14]. Also there is the problem of exon-exon junctions. By finding the longest increasing subsequence that problem can be tackled really well, as the algorithm will detect where those splice sites appeared.

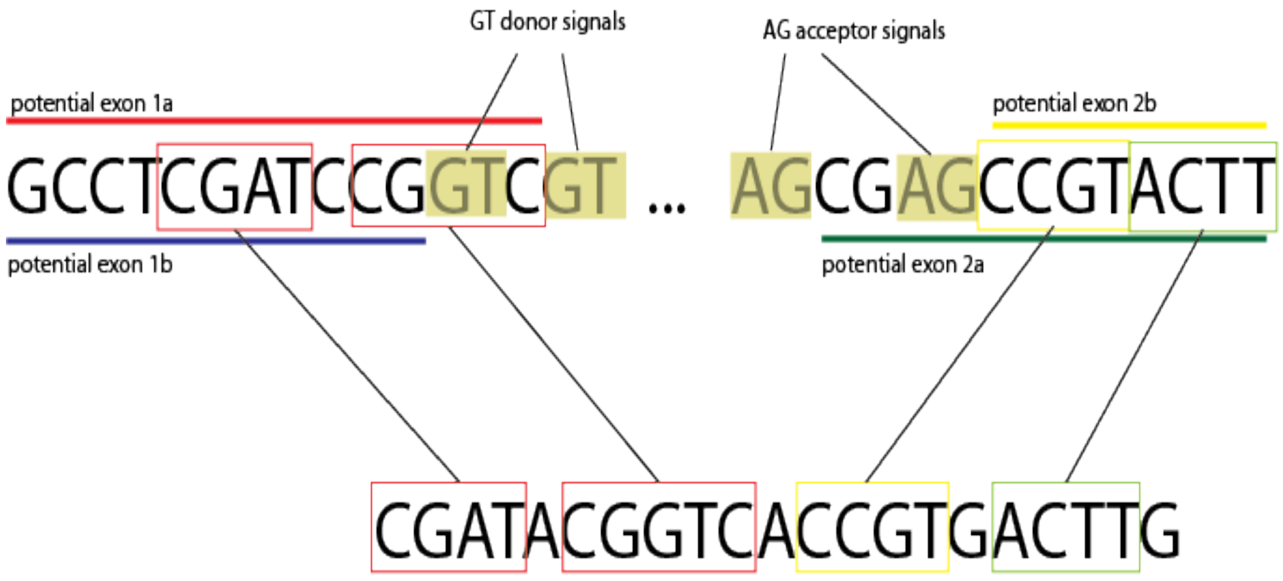

Figure 7. spliced alignment

Figure 7 shows an example of what can happen when mapping reads that span across exon-exon junctions. Potential exon regions were discovered in the previous phase of the alignment. There are four potential combinations of how the exon junction can look like. Potential exons are marked on the figure with red (potential exon 1a), blue (potential exon 1b), green (potentital exon 2a) and yellow (potential exon 2b). Also donor and acceptor signals were used in the previous phase to store potential splice sites that are now being examind. They are marked with yellow translucent rectangles – 'GT' signals right next to the end of potential exons 1a and 1b; also 'AG' signals next to the left end of the potential exon regions 2a and 2b. Below the genome is the read that is being mapped against its exon regions. After finding the longest increasing subsequence it turns out that this read supports splice junction of exon 1a and exon 2b. Rna-seq mapper will report occurences of other potential splice sites as well, it will grade exon-exon junctions that are supported by more reads better.

2.4.4. The index

It is shown that if the length of the seed is 16, more than 80% of seeds are going to be unique for the human genome [8]. This is the reason why this value is used in building up the index. Though, other values can be easily passed in as an option when the mapper is being called. First, over all the exon regions and potential splice site the index is being built. This is quite simple, a sliding window of chosen seed size is being swept through all the possible exon region resulting in an index [15].

G	T	G	A	C	C	T	C	A	G	G	T	C	G	T	G	A	C	C	T	C	G	T	A	G	T	C	A	G
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29

Figure 8. index over exon regions

ACCTCA	4
ACCTCG	17
AGGTCG	9
CAGGTC	8
CCTCAG	5
CCTCGT	18
CGTAGT	21
CGTGAC	13
CTCAGG	6
CTCGTA	19
GACCTC	3,16
GGTCGT	10
GTAGTC	22
GTCGTG	11
GTGACC	1,14
TAGTCA	23
TCAGGT	7
TCGTAG	20
TCGTGA	12
TGACCT	2,15

Figure 9. index with the sliding window of size 6

Figure 8 and *9* show the index built over the exon regions. A sliding window of size 6 was swept through the whole exons. Separated exon regions were all connected together before doing this step. For every chunk we have its position in the index.

Next step is running through all the reads that were unmapped in previous steps and trying to align them using the built index by using the measure of quality as the length of the longest increasing subsequence.

2.4.5. Longest increasing subsequence

Except the problem of insertions or deletions that common DNA aligners have, the problem of spliced reads in RNA sequencing is making things a little bit more complicated. As explained briefly before, it turned out that the length of the longest increasing subsequence can be a really good measure of quality for the reads that span exon-exon junctions. The basic idea is that the very ends of exons will be covered well when chopping the read in continous seeds of a given size. This smaller parts will be called seeds from now on. This approach gives also good results on seeds that map to whole exons, which is very important because this way reads that span even more than two exons can be discovered correctly.

For the sequence:

0, 7, 3, 11, 1, 9, 5, 13, 0, 8, 4, 12, 2, 10, 6, 14, ...

the longest increasing subsequence is:

0, 3, 5, 8, 12, 14

the best length that can be found for the given input sequnce is 6, the presented solution is not unique, though, there is one more increasing sequences of length 6:

0, 3, 5, 8, 10, 14

There are well known solutions for the problem of finding the longest increasing subsequence [16]. The solution that uses dynamic programming has quadratic time - O(n^2). This can be done more efficiently by using a data structure like Fenwick tree, by doing so the longest increasing subsequence can be found in

O(n log n) [17]. A Fenwick tree is a data structure that supports two operations on an array: increment a given value by a given amount and find the sum of segment of values. It can do both in O(log n) time. The Fenwick tree is represented as just an array of the same size as the array being updated and queried, and there is no need to store the original array itself. This is great because it means that the two mentioned operations can be supported without any additional memory.

After the longest increasing sequences have been found for the unmapped reads up to this phase, a report of matched and unmatched reads is being generated.

3. Implementation

The rna-seq mapper is implemented in the C++ programming language, mainly because of its efficiency, as one of the goals of this assignment was to build a tool that is comparable by performance, both accuracy and efficiency to the state-of-the art tools, such as Tophat or BBMap.

The whole implementation can be separeted into three main parts:

- Genome alignment
- Exon regions computation
- Spliced alignmen

3.1. Genome alignment

The input for this phase is the reference genome in the standard fasta format. Rna-seq is implemented in such way that it allows specifying before running the whole pipeline which aligner shall be used. Three options are provided here:

- BWA
- Bowtie (or bowtie2)
- LISA

Before doing the alignment, an index is built for each of those aligners. Each aligner handles its own format of index it uses. The output of this phase is the genome index (in the aligner internal format). After the index is built, the genome alignment phase starts. The overall final result of this phase is the aligned reads file in the standard SAM format.

3.2. Exon computation

The exon computation phase has two main inputs: the aligned reads in the SAM format from the previous phase and the refrence genome in fasta format. From the aligned reads file, potential exon regions are being computed as described in the methods section, by using various metrics like coverage, distance between exons, and heuristics like donor-acceptor signals. Except this file, the initial refrence genome file is needed as well, because for the next step, exon regions have to be extracted from the initial genome. This needs to be done in order to start spliced alignment, as at that point, reads are going to be matched against specific regions of the genome – exons.

3.3. Spliced alignment

Before spliced alignment there is a smaller phase that also falls into spliced alignment. The index has to be built from the exon regions. This index is needed as the initially unmapped reads are going to get mapped against it. Meaning, that this phase has two main input files: the exon index and the initially unmapped reads in fastq format. The index gets built by running the sliding window through the exon regions of the genome. There are four characters that appear in the genome: *'A'*, *'C'*, *'G'* and *'T'*. As the optimal value for the sliding window is 16, it turns out that the hash table size can fit into standard integer types for values below 16, or long long for slightly larger. After the index is built, the sliding window is being swept through all the unmapped reads that are waiting to be aligned. By running the sliding window, substrings of the reads (seeds) – their position on the exon genome is being extracted from the hash table (the exon index). After the possible mapping positions are being extracted for the read that is being currently processed, the longest increasing subsequence is being calculated. It is described in the methods section how the Fenwick tree data structure is used for better performance – both memory and running time. The result of this phase is the final report. It consists of two files: the aligned reads file in bam format and the junctions file, also in bam format. The alignment file shows for each aligned read where they mapped against the initial refrence genome. The junction file shows exon-exon junctions discovered by the tool.

3.4. The pipeline

Figure 10 shows the whole pipeline with all its main phases and subphases. It also includes all the main input files and resulting outputs of each step.

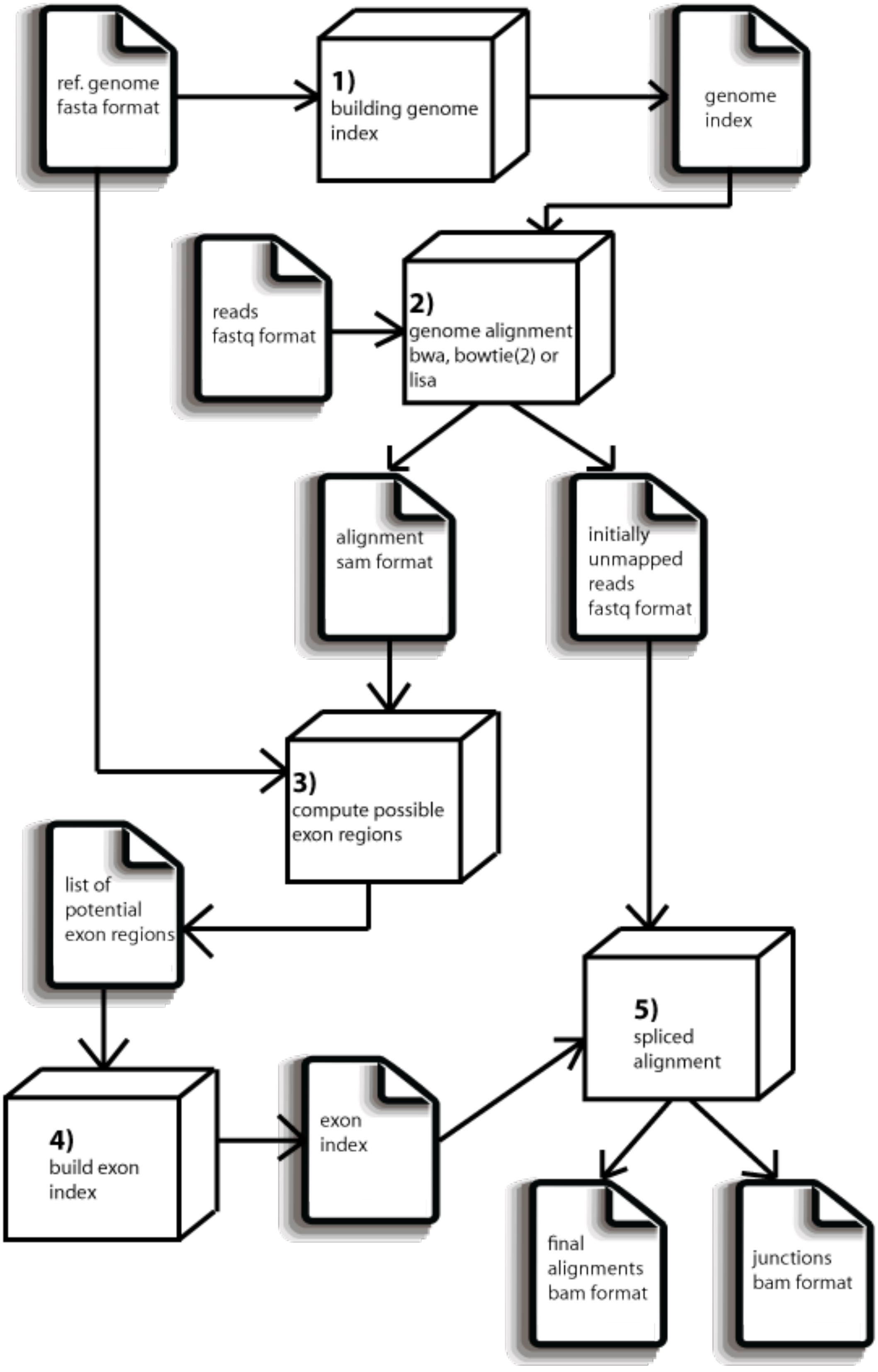
ref. genome fasta format
1) building genome index
genome index
reads fastq format
2) genome alignment bwa, bowtie(2) or lisa
alignment sam format
initially unmapped reads fastq format
3) compute possible exon regions
list of potential exon regions
4) build exon index
exon index
5) spliced alignment
final alignments bam format
junctions bam format

Figure 10. rna-seq pipeline

3.5. Phases analysis

In this section runtime analysis of the main phases of the rna-seq mapper are given. For this analysis 100000 randomly generated reads of length 250 (standard Illumina read length) with error rate of 0.5% that map onto chromosme 1 of human genome are being used.

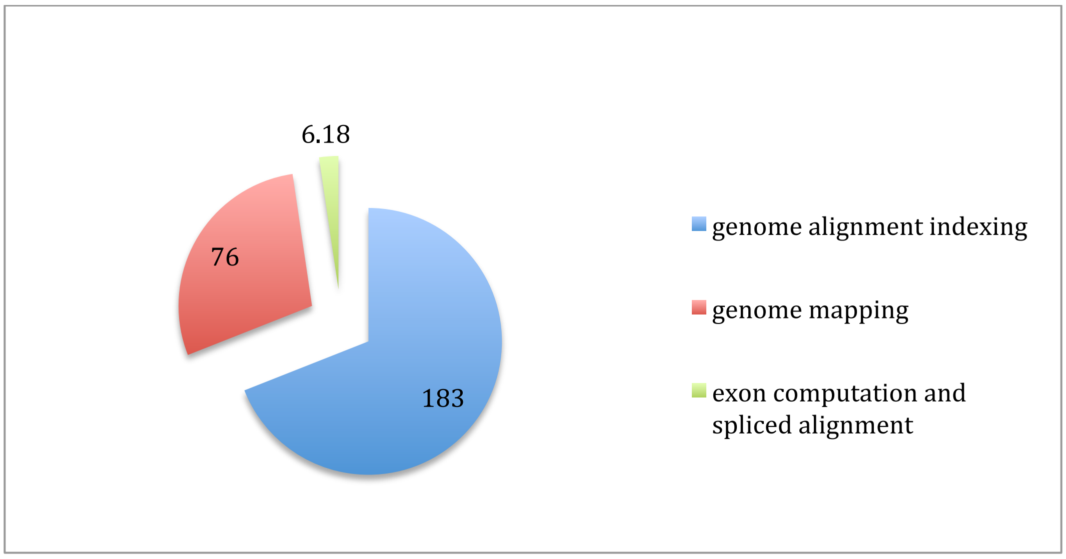

Figure 11. overall runtime analysis

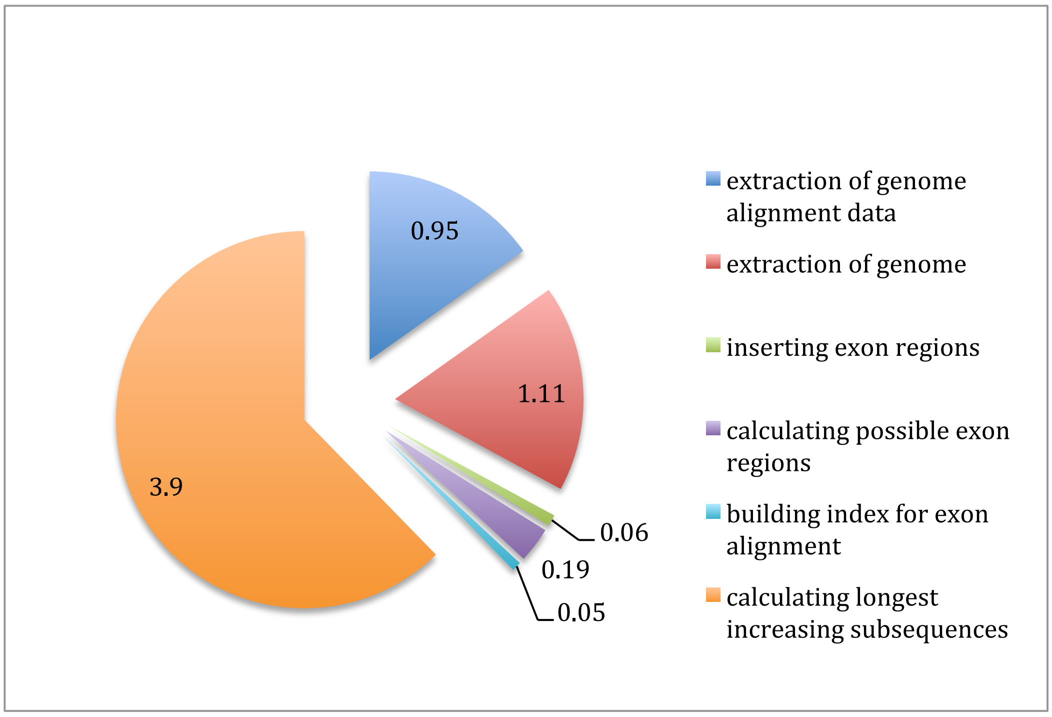

Figure 12. Spliced alignment and exon computation runtime breakdown

Figure 11 shows runtime values for major phases like genome index building, genome alignment and spliced alignemnt. Genome index building is obviously the slowest part of the process, but also most of the results discussions omit this part when comparing performances with other mappers. This step will be considered as preprocessing in this thesis as well, and is not going to be processed for calculating efficiencies of the tools. The genome alignment phase also depends on the short aligners that rna mappers use, so that part is also not getting into the final evaluation of performances.

The spliced alignment is the core of any rna-seq mapper as it tackles the main problem of rna sequenced data – the problem of spliced reads. *Figure 12* shows the runtime analysis of the subphases used in the new rna-seq mapper. The calculation of longest increasing subsequnce is the longest part and takes more than 60% of the spliced alignment phase. This is expected as it is the main algorithm for calculating quality of spliced mappings. In order to make this step faster, multithreading is being used. Standard C++ *pthread* library is being used in the implementation of multithreading.

4. Results

In this section performance of the new rna-seq mapper is being analyzed. The analysis was conducted on simulated data. The simulated data was created with the Beers simulator because of various reasons that are given in the next section. Rna-seq mapper is being compared with Tophat, BBMap and GSNAP, both over accuracy and time efficiency. All the tests have been executed on a sample of 100000 randomly generated reads.

4.1. Simulated data

For generating the simulated data Beers simulator (Benchmarker for Evaluating the Effectiveness of RNA-Seq Software) is being used. The main reason why this simulator was chosen over others is that it uses many different sets of annotation that were merged (some of the most prominent like AceView, Ensembl, Geneid, Genscan, NSCAN, RefSeq, SGP, Transcriptome, UCSC and Vega) [18]. To avoid biased dataset towards any particular annotation set, the simulator starts with a large number of gene models taken from about ten different published annotation sets. After that it chooses a fixed number of these genes at random and introduces substitutions, indels, alternate splice forms, sequnecing errors and intron signals. It can simulate both mouse and human data. Though, only human simulated data is being used for testing the rna-seq mapper. The simulator creates many files out of which four are most important. The .cig file has all the true alignments, this is very important because after the final results of the mapping of rna-seq mapper, with this file the results can be precisely compared. Knowing for each read from which exact location it was extracted is the main thing in order to evaluate the results properly. The .fa file has the actuall simulated paired-end reads. Forward reads are marked as 'a' reads, while reverse reads are marked as 'b' reads. There is also a file that has all the information for each read which crossed a junction in order to calculate the false negative rate, as junctions that were not crossed with any reads shouldn't be marked as junctions. And the simulated reads transcript file that has the full transcript info for each simulated transcript.

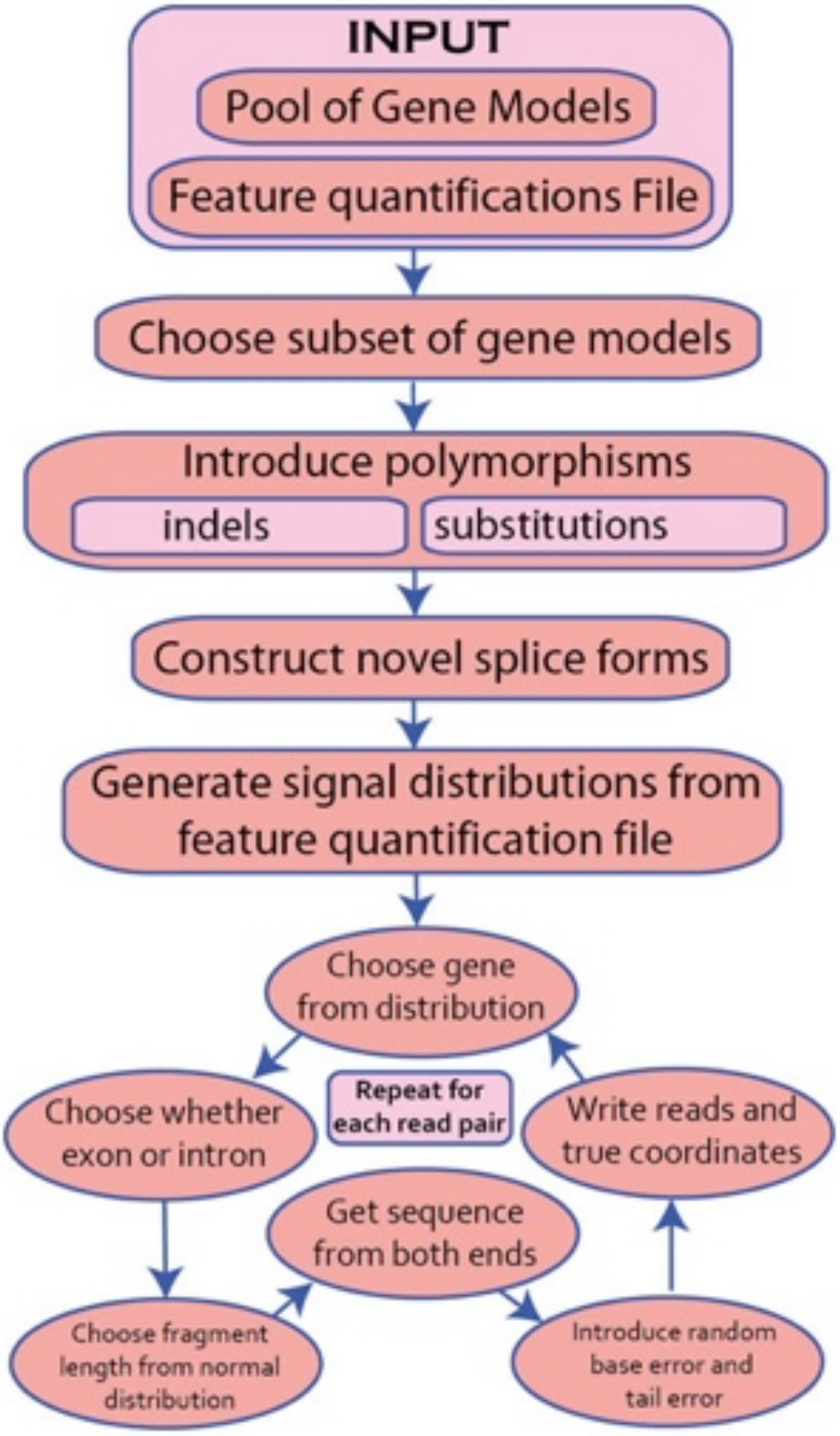
INPUT
Pool of Gene Models
Feature quantifications File
Choose subset of gene models
Introduce polymorphisms
indels
substitutions
Construct novel splice forms
Generate signal distributions from feature quantification file
Choose gene from distribution
Choose whether exon or intron
Repeat for each read pair
Write reads and true coordinates
Get sequence from both ends
Choose fragment length from normal distribution
Introduce random base error and tail error

Figure 13. Beers simulator workflow, image taken from [18]

4.2. Dataset runs

All the tests that follow have been generated with the described beers simulator. All of the tests consist of 100000 reads that were randomly generated from the human genome (hg19 – human genome, version 19). More than hundred tests have been run with 4 rna-seq tools:

- Rna-seq tool developed in this thesis
- Tophat2
- BBMap
- GSNAP

The results and the comparison between the tools for both time efficency and accuracy follow in next sections.

4.2.1. Dataset 1

The first dataset consists of 100000 randomly generated reads with an error of 0.5%. Reads of length 50, 75, 100, 250 and 500 were generated.

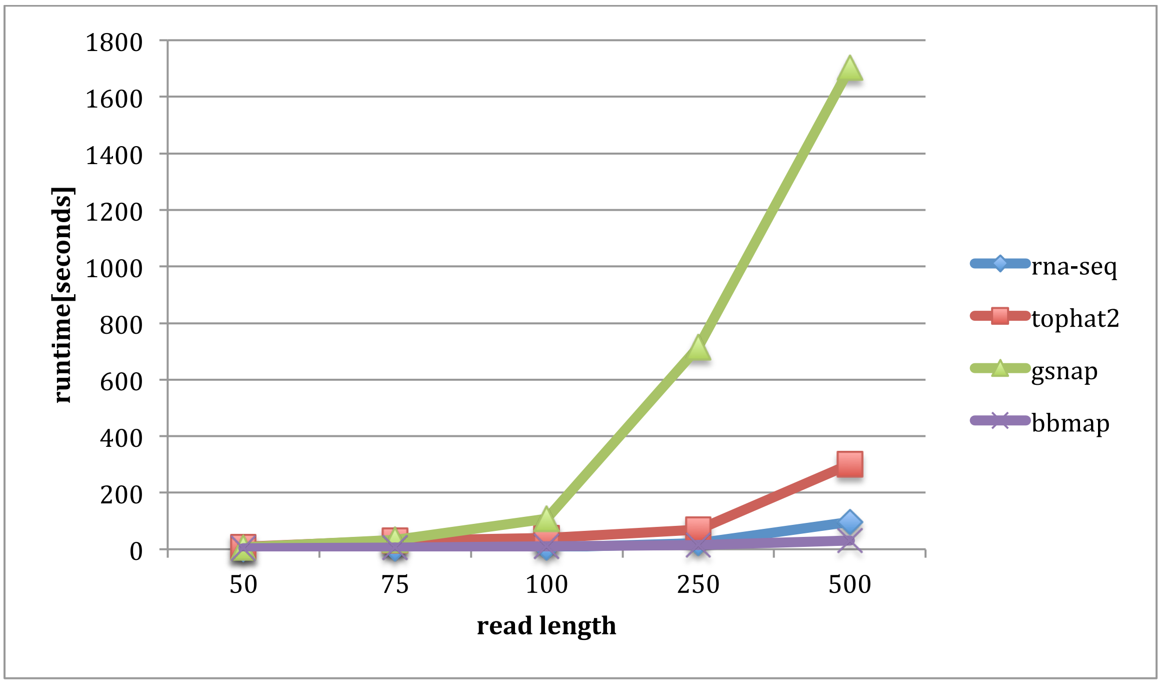

Figure 14. Time efficiency for reads with error of 0.5%

The fastest mapper on this dataset is BBMap. The new rna-seq tool is slightly slower than BBMap. It is faster than Tophat, and as the read lengths increase this difference becomes more visible. GSNAP is obviously the slowest. Because of this, a graph without GSNAP is given below as well, for a better comparison of the faster mappers.

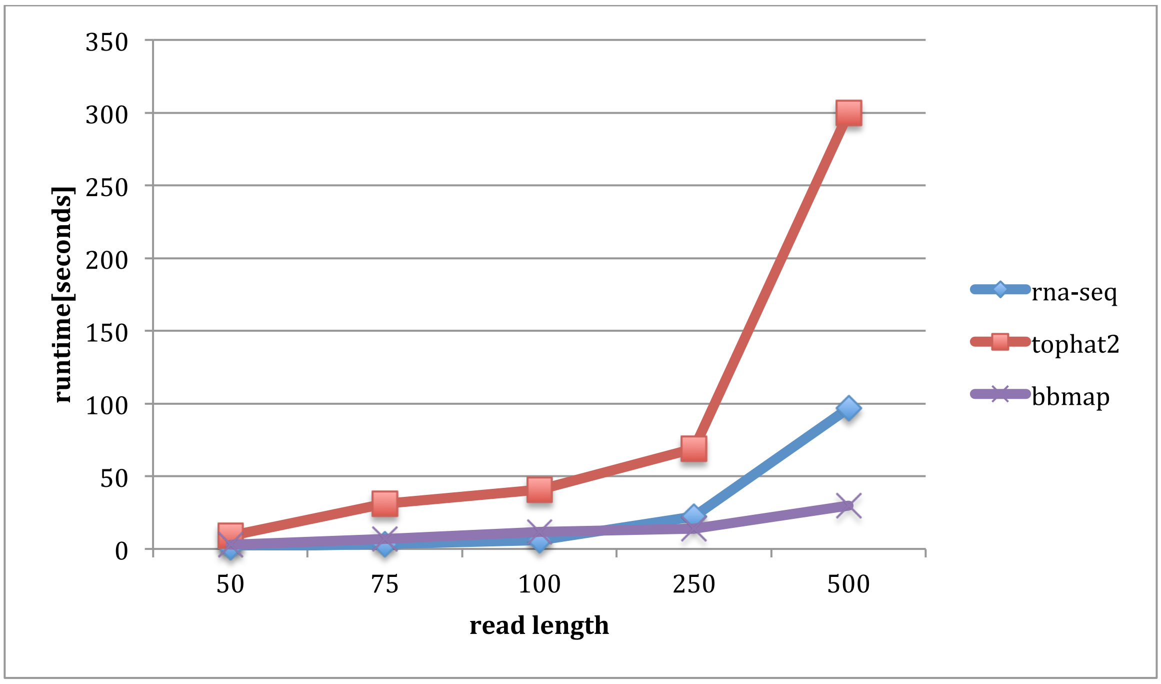

Figure 15. Same dataset without GSNAP

Read length	rna-seq	tophat	GSNAP	BBMap
50	99.5%	99.5%	97.45%	99.9%
75	93.25%	98.9%	99.85%	99.86%
100	89.29%	92.5%	99.7%	99.95%
250	82.87%	82.4%	99.8%	99.9%
500	61.29%	-	-	99.9%

Figure 16. Mapping accuracy for dataset 1

Rna-seq mapper is comparable with Tophat for reads below 250 length, after 250, rna-seq mapper becomes more accurate than tophat, while staying faster than it. For read lengths of 500, Tophat and GSNAP haven't executed, yielding inernal errors, after multiple tries to execute it. BBMap and GSNAP are superbly accurate above Tophat and rna-seq mapper. Though, GSNAP has an exponential growth in time and for larger sets it does not get executed at all. BBMap has great results, both accuracy and time efficiency.

4.2.2. Dataset 2

This dataset has a slightly increased error rate of the reads, it is 1%.

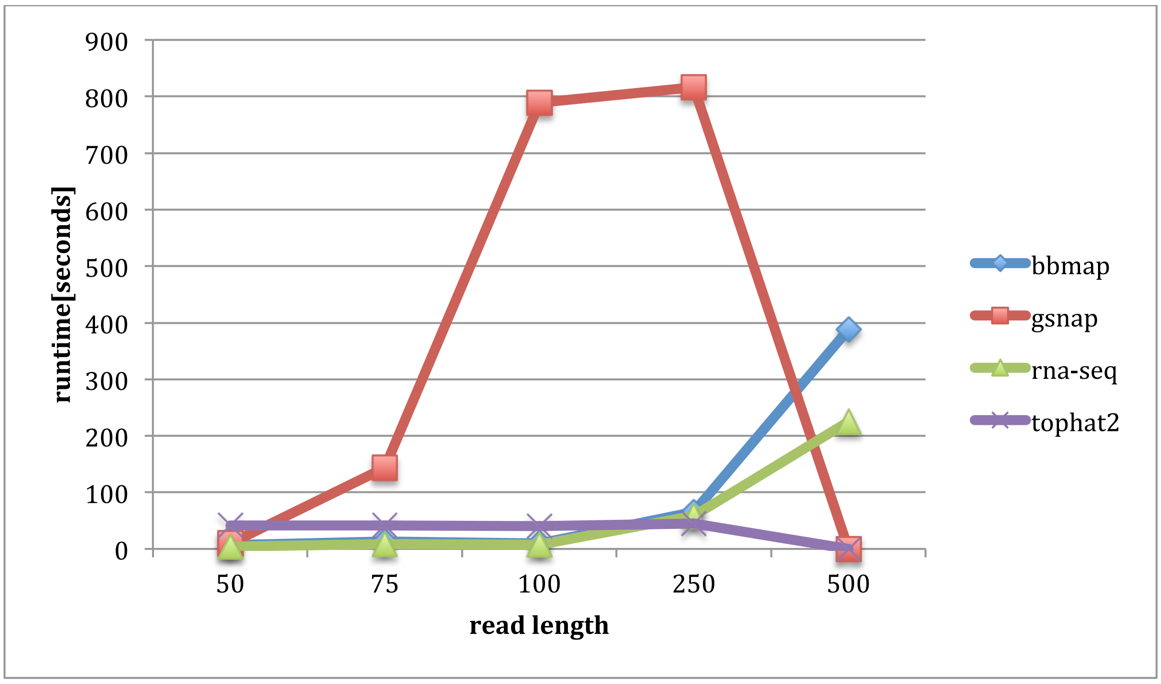

Figure 17. time efficiency comparison

GSNAP and Tophat didn't execute for read length of 500. More interesting on this graph are runtimes for BBMap and rna-seq. Next chart gives a better look on those two mappers.

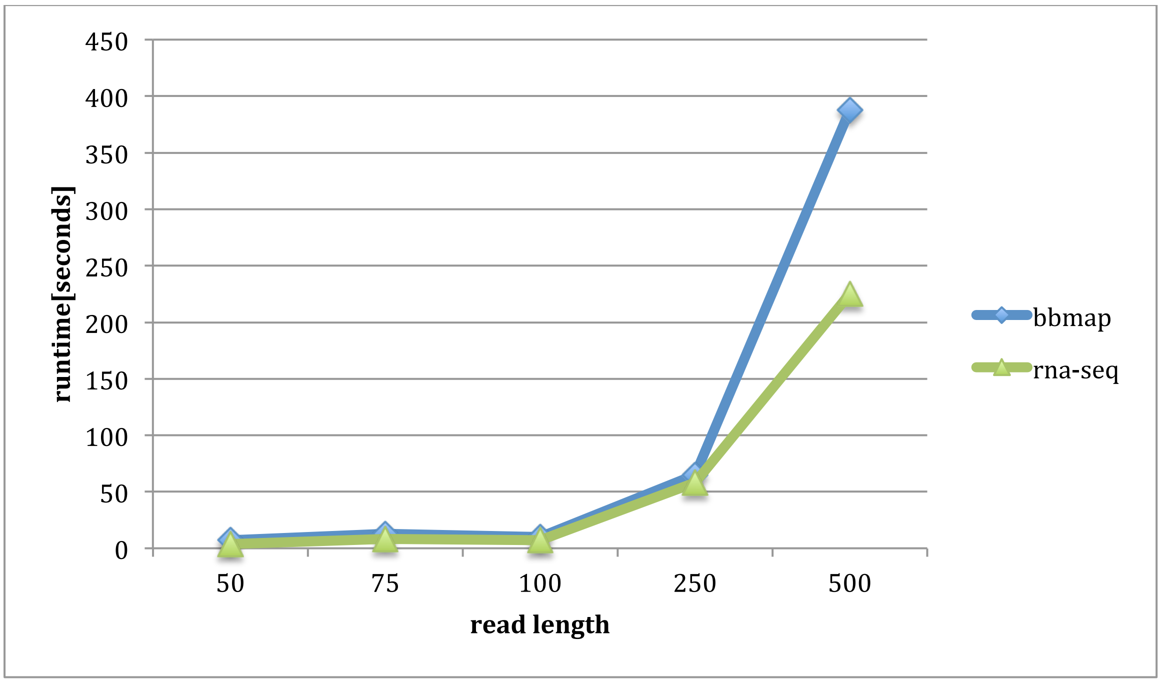

Figure 18. bbmap vs rna-seq

Read length	rna-seq	tophat	GSNAP	BBMap
50	82.5%	96.2%	78.03%	99.84%
75	77.83%	94.5%	99.2%	99.82%
100	81.48%	90.9%	99.5%	99.39%
250	58.59%	49.6%	99.7%	99.9%
500	34.29%	-	-	90.7%

Figure 19. mapping results for dataset 2

Rna-seq has lower accuracy than Tophat, until reads of length 250 are reached when it becomes slightly more accurate but still less than BBMap and GSNAP. GSNAP and Tophat haven't executed for read lengths longer than 250 for this datasets. BBMap finally drops its accuracy below almost perfect score for reads that have 1% error and length 500.

4.2.3. Dataset 3

In this dataset read error has been increased to 2.5%. For non obvious reasons GSNAP stopped executing for this dataset so GSNAP is omitted in this analysis.

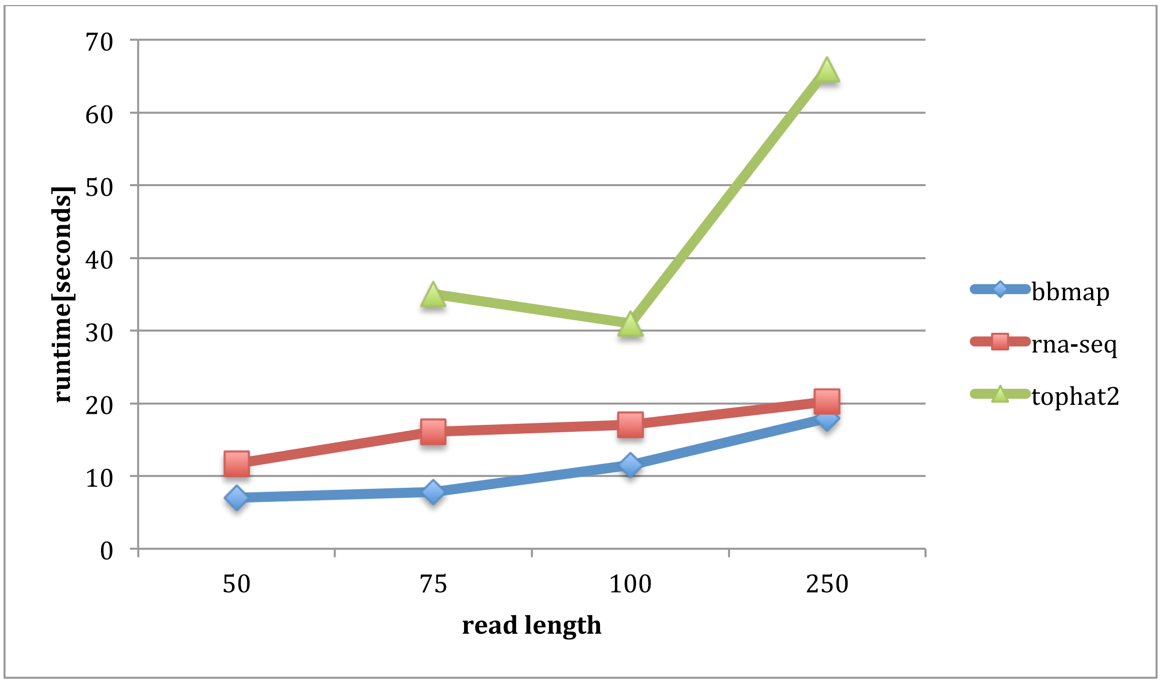

Figure 20. runtimes for dataset 3

Read length	rna-seq	tophat	BBMap
50	66.46%	-	99.9%
75	58.9%	51.7%	99.82%
100	58.6%	53.4%	99.2%
250	60.2%	52.6%	99.1%
500	55.21%	-	-

Figure 21. accuracy comparison for error rate 2.5%

Rna-seq mapper for reads with an error rate of 2.5% is more accurate than Tophat and also faster, still BBMap is far better. Tophat didn't execute on this set for read lengths of 50 and 500, neither did BBMap for read length of 500.

4.2.4. Dataset 4

Error rate of the reads got increased to 5% this time, but only comparison between rna-seq mapper and Tophat are given this time as other mappers had difficulties running it.

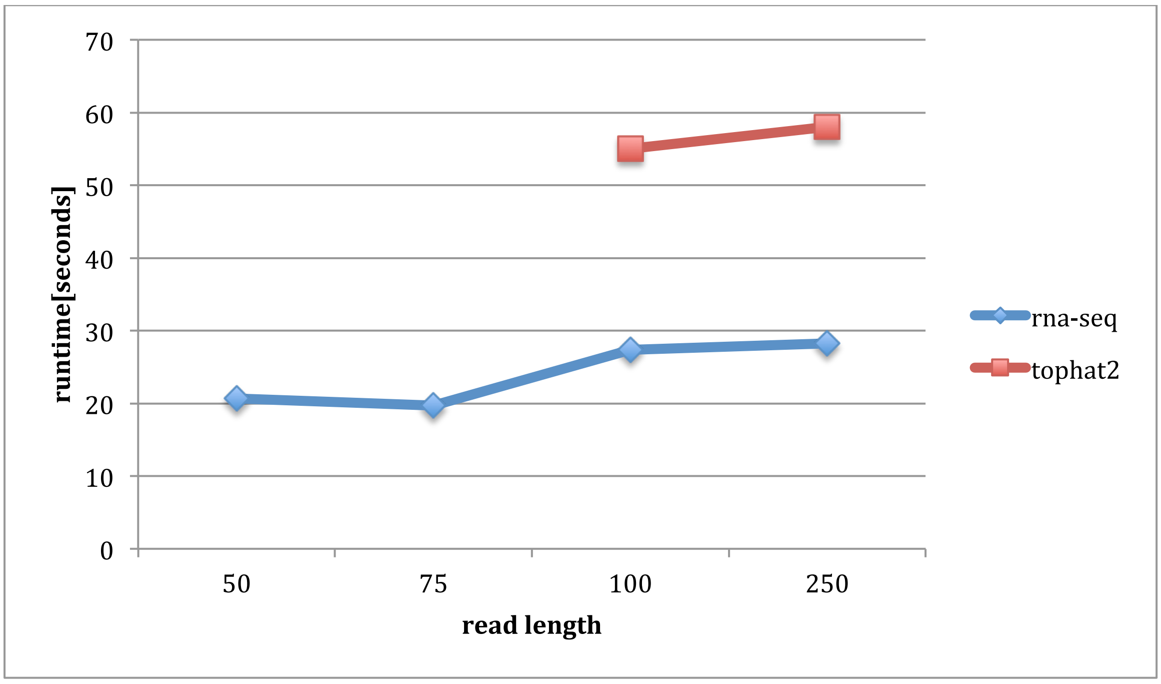

Figure 22. runtimes for dataset 4

Rna-seq mapper is again faster than Tophat, accuracies for both mappers are considerably low. This is mainly because of the high error rate of 5% of the reads. Still, rna-seq mapper is more than twice accurate than Tophat, with about 28% reads mapped correctly, while Tophat mapped only about 11% reads. This result is shown on the next chart.

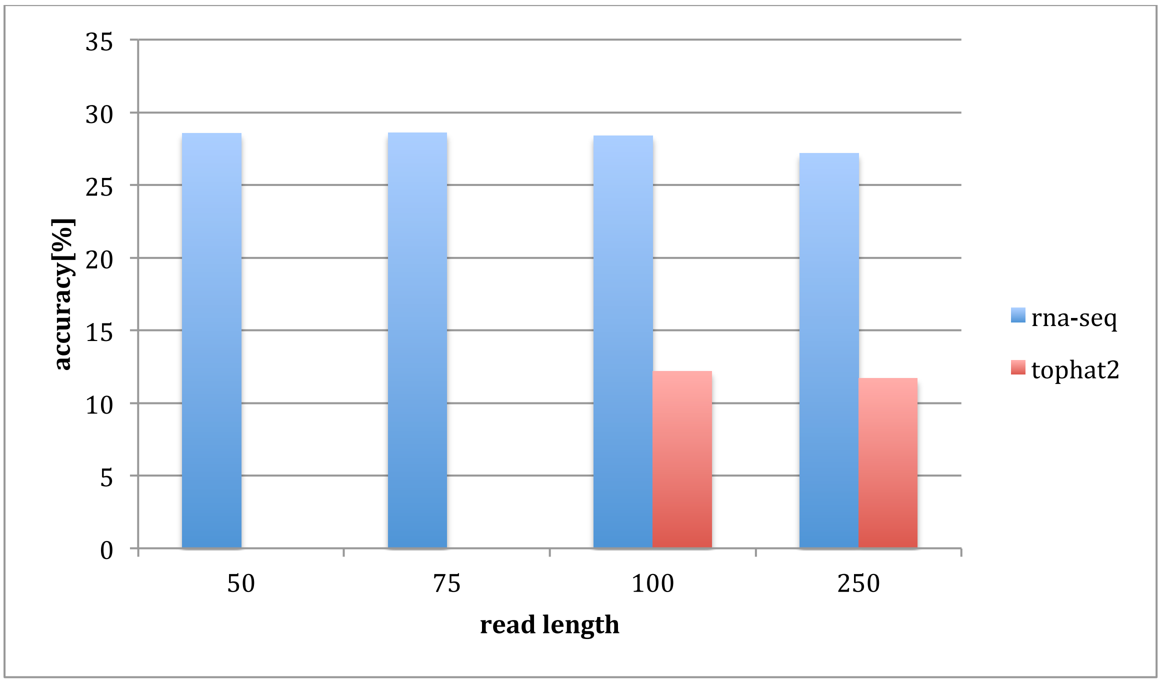

Figure 23. accuracy for dataset 4

4.2.5. Dataset 5

This dataset has an extreme read error of 10%.

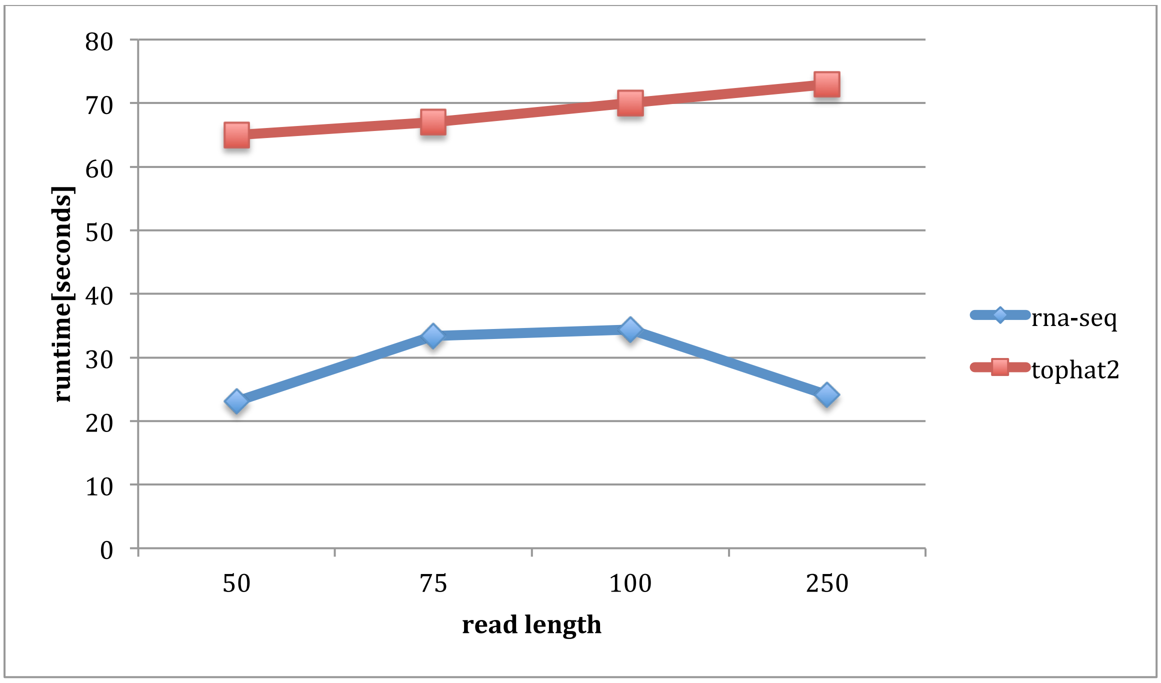

Figure 24. runtimes for reads with 10% error

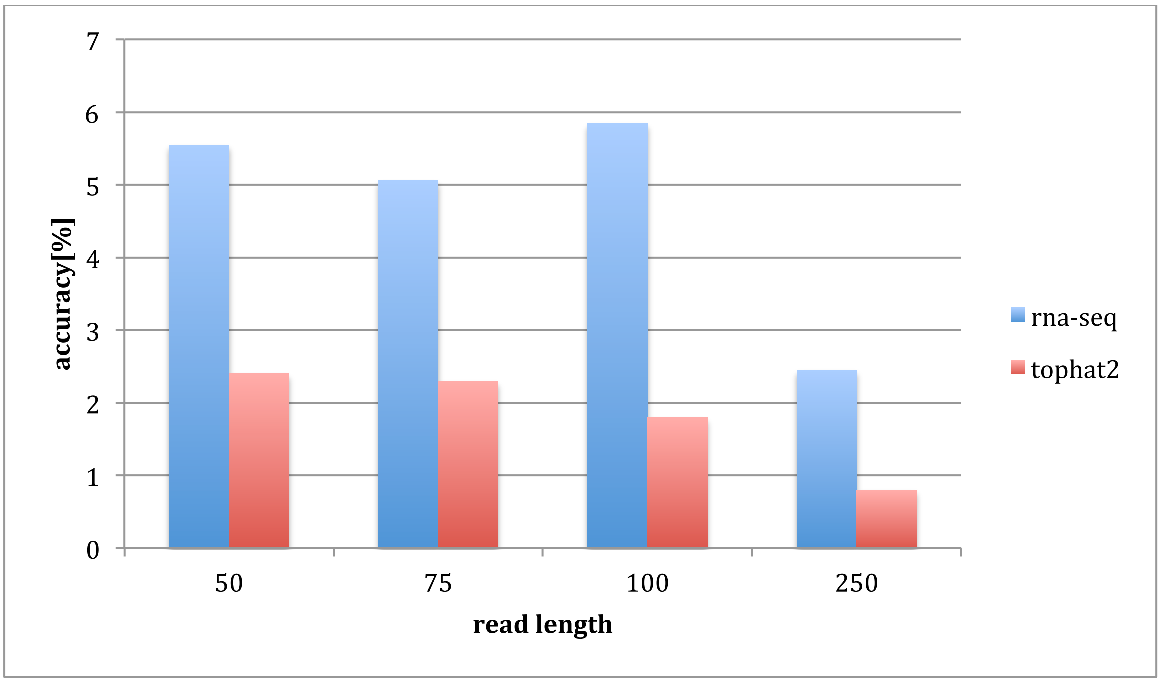

Figure 25. mapped reads with 10% error rate

Figure 24 and *25* show runtime and accuracy comparison for reads with an extreme error rate of 10%. Rna-slseq is both faster and more accurate than Tophat on this dataset. Though both mappers have a really low accuracy (rna-seq about 5% and Tophat below 2.5%).

4.2.6. Overall rna-seq scores

In this section comparison between scores for rna-seq mapper are given for each of the five datasets that differ in the reads error rate. As expected, as the error rate increases, so does the accuracy of alignments fall. The runtime for spliced alignment increases as the reads length increase as expected. This is mainly because more seeds are being examined for the calculation of the longest increasing subsequence. The results are given on *Figure 26* and *27*. *Figure 27* shows the same input data runs, but for Tophat.

	0.5%	**1%**	**2.5%**	**5%**	**10%**
50	99.5%	82.5%	66.46%	28.58%	5.55%
75	93.25%	77.83%	58.9%	28.6%	5.06%
100	89.29%	81.48%	58.6%	28.4%	5.85%
250	82.87%	58.59%	60.2%	27.2%	2.45%
500	61.29%	34.29%	30.1%	24.2%	1.86%

Figure 26. rna-seq accuracy for various read errors and lengths

	0.5%	**1%**	**2.5%**	**5%**	**10%**
50	99.5%	96.2%	n/a	n/a	2.31%
75	98.9%	94.5%	51.7%	n/a	2.23%
100	92.5%	90.9%	53.4%	12.2%	1.86%
250	81.9%	49.6%	52.6%	11.7%	0.76%
500	n/a	n/a	n/a	n/a	n/a

Figure 27. tophat2 accuracy for various read errors and lengths

Figure 26 and *27* are showing a comparison between accuracies for the new rna-seq against Tophat. As the error rate and read length increase rna-seq gets better against Tophat. *Figure 28* shows that read error rate did not impact much on the performance, that is expected as the main parameters for efficiency are genome size, read amount and read length, but not read errors.

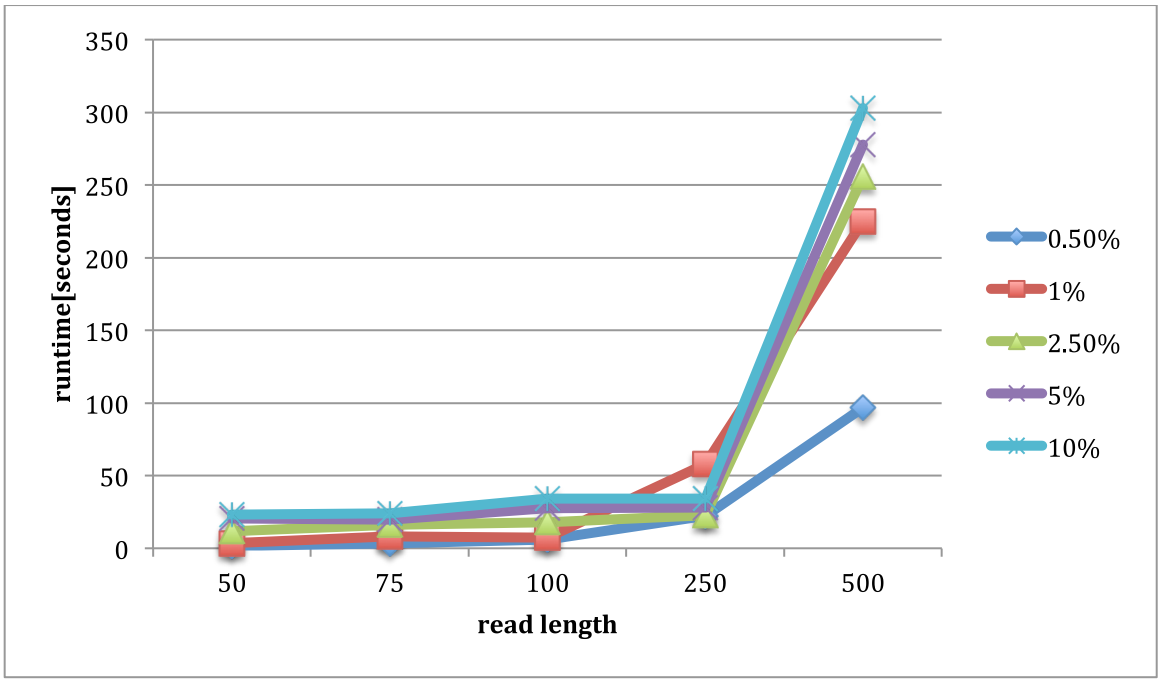

Figure 28. rna-seq runtimes for various read errors

Figure 29 and *30* show percentage of correctly mapped reads after the main phases of the rna-seq mapper, genome and spliced alignment. As the read lengths increase genome alignment has lower accuracy. This is expected because most of the exons are considerably short, and as read lengths increase, some exon regions do not get detected before spliced alignment. This also means that spliced alignment works well, as it maps more reads on spliced alignment phase on longer read lengths, meaning that longest increasing subsequence measure of quality for spliced reads has potentials.

	50	**75**	**100**	**250**	**500**
Genome alignment	99.51%	90.39%	82.05%	74.43%	29.09%
Spliced alignment	99.53%	93.25%	89.29%	82.87%	61.29%

Figure 29. percentage of mapped reads after main phases for rna-seq mapper for reads with error rate of 0.5%

	50	75	100	250	500
Genome alignment	80.54%	69.07%	72.48%	34.46%	6.21%
Spliced alignment	82.5%	77.83%	81.48%	58.59%	34.29%

Figure 30. percentage of mapped reads after main phases for rna-seq mapper for reads with error rate of 1%

4.2.7. Additional notes

Here are the commands that were used for running the datasets:

rna-seq	rna-seq bowtie2 genome.fa idxPrefix reads.fq res.sam > summary
tophat	tophat2 idxPrefix reads.fq
BBMap	bbmap.sh ref=genome.fa in=reads.fq out=bbmap.sam
GSNAP	gsnap –A sam –N 1 –d genome.fa reads.fq > gsnap.sam

Figure 31. commands to run the mappers

rna-seq	this step is done automatically
tophat	bowtie2-build genome.fa idxPrefix
BBMap	this step is done automatically
GSNAP	gmap_build –d genome.fa genome

Figure 32. commands for building indexes for the mappers

For testing the accuracy a script called bedVsBeers was written. It compares rna-seq mapper output that is in bed format against the beers simulator cig file that contains information for all generated reads where are they mapping on the genome. This script can be run with the command 'bedVsBeers <aln.cig> <aln.bed>'. It gives feedback about how much of the reads mapped correctly against the initial beers simulator reads positions. Some of the tested mappers such as BBMap, Tophat and GSNAP yield results in bam or sam formats. Those formats can be easily converted to bed using samtools and bedtools.

The whole rna-seq mapper project can be found on:

https://github.com/ijerkovic/rna-seq

5. Conclusion

In this thesis a new spliced rna-seq mapper was developed and presented. Comparison between some popular spliced aligners like Tophat, BBMap and GSNAP against the new rna-seq mapper was given. It has been shown that in its early phase of development the rna-seq mapper has good performances, especially over Tophat which was the main tool for comparison. Those differences are becoming clearer as the error rate of the reads increase, and as the read lengths increase. Nevertheless, a lot of work still needs to be done as BBMap and GSNAP proved to be more accurate. Though, GSNAP gets this precision at a large cost of exponential runtime. The most interesting thing compared to Tophat is that the new rna-seq mapper has better results than it when read lengths increase. This could mean that the longest increasing subsequence as a measure of quality for spliced alignment has a lot of potential. It is inevitable that by increasing the read lengths, the overall accuracy of spliced mappings fall. Primarily as the read lengths go above the exon lengths, some exon regions never get detected by the genome alignment phase, losing the potential true alignments for next phases. This has a potential for future work, as it could be overcome by slicing the initial reads into smaller parts for exon search. After the smaller parts have been mapped on the genome, exons can be reconstructed and the pipeline can continue. But this has to be yet proved in real dataset experiments. As of implementation priorities, the thread pool implementation should be improved, or even some finished solutions could be reused. As of implemented options for running the tool, currently there are the most essential ones, but for making more custom experiments, more customized options shall be allowed in the future.

6. Literature

[1] R. D. Morin et al: *Profiling the HeLa S3 transcriptome using randomly primed cDNA and massively parallel short-read sequencing*, 2008

[2] C. A. Maher et al: *Transcriptome sequencing to detect gene fusions in cancer*, 2009

[3] J. T. Witten, J. Ule: *Understanding splicing regulation through RNA splicing maps, 2011*.

[4] D. Kim et al: *Tophat2: accurate alignment of transcriptomes in the presence of insertions, deletions and gene fusions*, 2013

[5] C. Trapnell: *Tophat: discovering splice junctions with RNA-seq*, 2008

[6] B. Bushnell: BBMap short read aligner, URL http://sourceforge.net/projects/bbmap/

[7] K. Wang: MapSplice: Accurate mapping of RNA-seq reads for splice junction discovery, 2010

[8] Y. Liao: The Subread aligner: fast, accurate and scalable read mapping by seed-and-vote, 2013

[9] T. D. Wu, C. K. Watanbe: GSNAP: a genomic mapping and alignment program for mRNA and EST sequences, 2005

[10] A. Dobin et al: STAR: ultrafast universal RNA-seq aligner, 2012

[11] S. S. Ajay: *Accurate and comprehensive sequencing of personal genomes*, 2011

[12] A. Y. Karpova et al: *Dual utilization of an acceptor/donor splice site governs the alternative splicing of the IRF-3 gene*, 2000

[13] G. Rautmann: *Synthetic donor and acceptor splice sites function in an RNA polymerase B (II) transcription unit*, 1984

[14] F. Pavetic, G. Zuzic: LISA – DNA reads alignment tool, 2013

[15] R. Sedgewick: Algorithms, 2003

[16] C. Schensted: *Longest increasing and decreasing subsequences*, 1961

[17] P. Mitrichev: *Fenwick tree range updates*, 2013

[18] G. R. Grant et al: Comparative *Analysis of RNA-Seq Alignment Algorithms and the RNA-Seq Unified Mapper (RUM),* 2011

RNA-seq mapper – a tool for mapping RNA sequenced data

Abstract

RNA sequencing is a method for transcriptome analysis. Understanding the transcriptome is essential for various reasons such as interpreting the functional elements of the genome and undrestanding development of diseases, among many other use cases. This thesis presents a new RNA-seq spliced alignment algorithm, that uses finding the longest increasing subsequence in detecting splice sites and mapping spliced reads across exon junctions. Comparison between some of the state-of-the-art tools such as Tophat and BBMap are given.

Keywords: RNA sequencing, hash, longest increasing subsequence, bioinformatics, genome, transcriptome

RNA-seq mapper – alat za mapiranje podataka dobivenih RNA sekvenciranjem

Sažetak

RNA sekvenciranje je metoda za analizu transkriptoma. Razumijevanje transkriptoma je izuzetno važno iz nekoliko razloga kao što su shvaćanje funkcionalnih elemenata genoma i razumijevanje razvoja pojedinih bolesti, što je samo par od mnogih mogućnosti primjene. Ovaj rad prikazuje novi algoritam za mapiranje podataka dobivenih RNA sekvenciranjem, posebice za očitanja koja se prostiru preko više eksoma. Usporedba sa trenutno najhvaljenijim i najboljim sličnim alatima kao što su Tophat i BBMap prikazana je u radu.

Ključne riječi: RNA sekvenciranje, hash, najdulji rastući podniz, bioinformatika, genom, transkriptom