**MGMM – A Bowtie based mapper for Oxford Nanopore Sequencing reads**

**Ron Fechter, Jonas Galli,**

**Felix Offensperger, Florian Tichawa**

**Created for GoBi 2017**

**LMU**

**Prof. Zimmer**

Inhaltsverzeichnis

1. General scheme of existing mappers 3

2. Seeding 4

3. Filtering and Chaining 4

4. Alignment 5

4.1 Start and end position estimation 5

4.2 Pathing 5

4.3 Alignment 6

4.4 Stitching 6

4.5 Multithreading 6

5. Analysis of seeds and chains 6

# 

# Abstract

*The Oxford Nanopore Sequencing Technology creates very long, error-prone reads with a length of roughly 1000bp and an average sequence identity of 70% and InDels due to technical reasons. These reads open new possibilities for genetic research fields like genome assembly and alternative splicing analysis. The downside of these atypical reads is that common mappers for short reads are unable to map these large reads within acceptable runtime. There are multiple special mappers to serve this purpose, we will concentrate on BWA, Last and GraphMap for this paper.*

*Unfortunately, only GraphMap offers the possibility to map these reads in transcriptomic space, and the results don't seem to differ from genomic mode (While ignoring introns).*

*This is why we decided to build our own MinION specialized transcriptomic mapper. MGMM is based on Bowtie and focuses on finding new splice variants from the known splicing sites.*

# EINLEITUNG [RON]

# 1. General scheme of existing mappers

To be able to handle the requirements of nanopore sequencing mapping, new properties of read-to-reference alignment had to be considered to result in a stable sensitivity and recall for very long reads with a low identity. Typical for these alignments are programs such as BLAST, which provide high sensitivity, but become infeasible for larger genomes for example the human genome. Differently to that BWA-MEM and BLASR provide a different tradeoff, scaling well to large genomes but have lower sensitivity and a low precision for high-error rate reads. This can be explained by the specific parameter settings of BLASR, which are designed for PacBio reads and can be chosen when using BWA-MEM for different sequencing technologies. DALIGNER, a highly sensitive overlapper which additionally supports read mapping, also provided precision and recall that degraded quickly with read error rate and genome size. Other mappers, such as LAST, are originally designed for aligning genomes, traded better with these setting, but still exhibits lower recall for large genomes.

Having a closer look on mapper that had a good performance in mapping read-to-reference on a larger genome, approaches of LAST, BWA-MEM and GraphMap for seed finding, seed extension and determining regions on the reference genome seem to be interesting and will be compared in the following section. As a first step of the paradigms of these mappers, region selection relies on finding seeds between the query sequence and the reference. Those can be distinguished by the scoring of the seeds, BWA-MEM uses maximal exact matches (MEMs) or supermaximal exact matches (SMEMs), LAST Hamming distance based spaced seeds and GraphMap a form of gapped spaced seeds, similar to gapped q-gram filters for Levenshtein distance. Fixed length seeds were found to be either not sensitive enough or not specific enough in the presence of high error rates (as used in LAST). BWA-MEM is reducing mismappings caused by missing seeds by re-seeding. The next step is filtering those seeds into candidate regions, with different criteria, like BWA-MEM is greedily chaining seeds that are co-linear and close to each other, and filtering short chains that are largely contained in a long chain and are much worse than the long chain to reduce unsuccessful seed extension in later steps. GraphMap is using for binning seed hits instead the concept of a Hough transform (HT), which corresponds to the main diagonal in the dynamic programming alignment matrix and is clustering the seeds into anchors under graph-based vertex-centric construction. For extending these anchors the Longest Common Substring of kmers (LCSk) had been built with variable lengths corresponding to the anchors, and are filtered with L1 regression to determine the region on the reference genome. A ranking of those anchors is made by the number of exact kmers covered by the anchor, the length of the query sequence, which matched the target, the number of bases covered by anchors and the read length. BWA-MEM got a similar approach of ranking those seeds, criteria are length of the chain it belongs to and then by the seed length, after dropping seeds that are contained in an alignment found before, those seeds are extended with banded affine-gap-penalty dynamic programming matrices. Later heuristics avoids extension through a poorly aligned region with good flanking alignment.

# 2. Seeding

The length of the reads, mean of 1000 bp, makes it impossible to map the complete sequence against the genome; therefore, the first step of the alignment is seeding. During this step, the read is fragmented into small parts called seeds. A fix shift of five bp and a fix length were detected when analysing the coverage and the loss of mappings. For simplicity, the read is seeded in the sense he is written in the *fastQ* file and the reverse complement of it, allowing so a mapping on the reverse strand too.

After the fragmentation, the seeds are mapped with Bowtie 2. We use Bowtie 2 because of its quickness and precision in mapping short reads against a long genome. The parameter for the mapping are End-to-End mapping, allowing a maximum of one mismatch, not allowing mapping on the reverse strand and allowing only 100 alignments for each seed.

# 3. Filtering and Chaining

After the Bowtie 2 mapping of the seeds, the mappings must be filtered and grouped. Since not all mapped regions are relevant, only for those with overlapping seeds a further mapping makes sense. The first filtering step is the deletion of unmapped seeds. Then groups of overlapping seeds are identified. For each seed, we search for overlapping mappings, if there is at least one overlapping seed and the seeds are in a correct order the overlap can be extended forming a seed chain. The detection of the right order occurs using the seed ID. Knowing the ID, the seed length and the used shift one can determine if there is an overlap between the two seeds on the read. Only if there is an overlap on the read the seeds are added to the chain, else they are filtered out. The chain will be extended until the last seed of the chain will not have any overlap with other seeds. If a seed has no other overlapping seeds but in between two seeds in the right order, the seed is annotated as well. After filtering out the mappings of the sense or the antisense, due to the number of mappings or, if similar, the length of the chains, all annotated chains are written to a so-called *fastM* file. In this file, the chains are grouped for all chromosomes of a read.

# 4. Alignment

In the first step the mapper creates a list of all genes containing all relevant seeds, calculates all possible start and end positions of the read based on all known transcripts and gathers these coordinates as fixed points for the alignment.

The second step is the pathing. The pathing algorithm determines all intron / exon combinations filling the gap between every pair of consecutive fixed points in transcriptomic and genomic space.

In the final step the aligner calculates a score for every pathing, selects the pathing with the highest score in every seed gap and creates the best possible alignment for every gene from the list created before.

## 4.1 Start and end position estimation

In most cases the first and last mapped seed from a given read don't cover the read start and end. Because of that the read start and end position have to be estimated before being able to create a pathing. The mapper tracks the transcriptomic position of the read start (or end) back (or forward) based on the first (or last) mapped seed, translates this position to a genomic coordinate for every known transcript and executes the following steps for every possible start (or end) position. During the stitching the algorithm also selects the best start (or end) position based on the respective alignment scores.

## 4.2 Pathing

The pathing is required because the gap between two seeds is still too big to be aligned as is. It's based on the assumption that every alignment consists out of close-to-linear exons (Besides the InDels resulting from the Nanopore Sequencing technology) and gap-free introns. Furthermore the algorithm assumes that every splice variant can be derived from the already known 5' and 3' splicing sites.

The pather creates a list of all possible introns and exons in the mapping area by combining every 5' with every 3' splicing site and creates all subsets from this list with a total transcriptomic length equal to the read length and a total intronic length equal to the difference between genomic and transcriptomic length of the mapping area.

Every subset now represents a rough mapping attempt between two seeds (Or a start / end position candidate and a seed).

## 4.3 Alignment

The aligner calculates a score for every pathing by aligning the exon parts to their respective genomic sequence (According to the pathing) with a slightly changed Gotoh algorithm in global (For gaps between two seeds), start-freeshift (For start position candidates) or end-freeshift (For end position candidates) mode.

## 4.4 Stitching

The final part consists of comparing all mapped pathings for every seed gap, picking the best-fitting pathing and inserting the intron and seed sequences to create the best possible alignment for every gene within the mapping area. The mapping with the highest score between all genes is marked as primary alignment, everything else is marked as secondary.

## 4.5 Multithreading

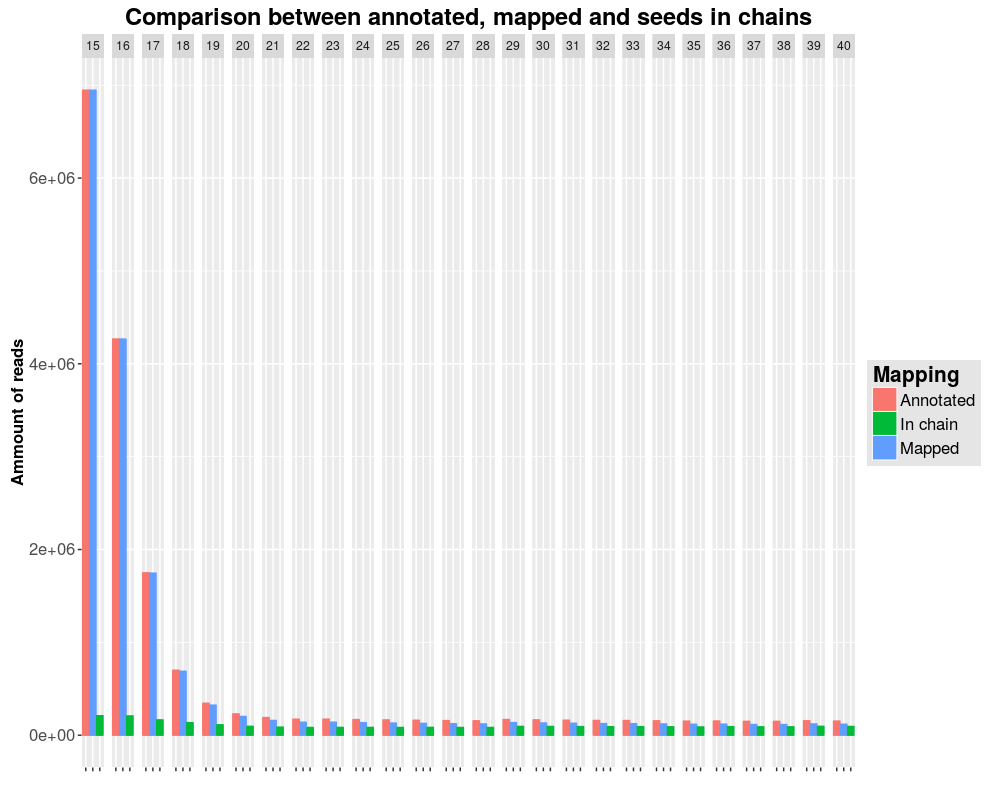
Due to absolute isolation of every read during the alignment it's very easy to distribute the reads to different threads to map them independent and faster. The Java implementation uses the Java 8 Stream API which can be used to split the entire alignment process into multiple threads. The only choke points remaining are input and output operations while reading fastM or writing SAM files.

# 5. Analysis of seeds and chains

In this part a comparison between different seed lengths and the consequences for the further steps is done. For the analysis a simulated test set of 466 reads was created. The parameters for the simulation were previously extracted from real MinION reads and result to be:

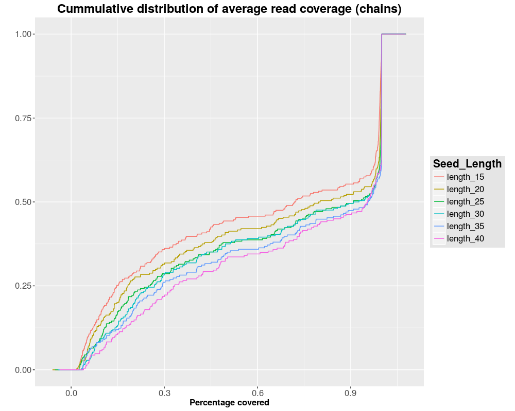
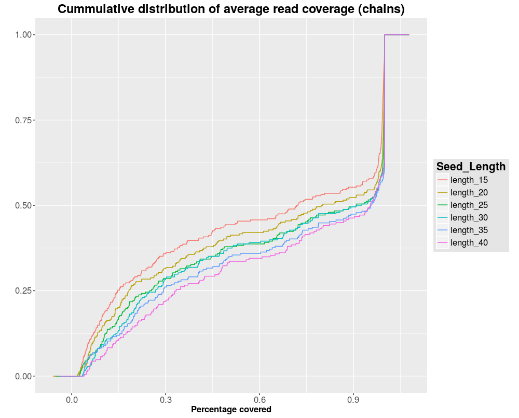
* Mean read length: 1000 bp
* Read standard deviation: 100 bp
* InDel rate: 1%
* Error rate: 30%

The first comparison simply shows the effect of the seed length on the amount of annotated alignments after the bowtie mapping. As evidenced in FIGURE? shorter reads, obviously, behave much more mappings on the complete genome then longer ones, but when looking only to those that are not filtered out during the chaining step the differences are not that evident.

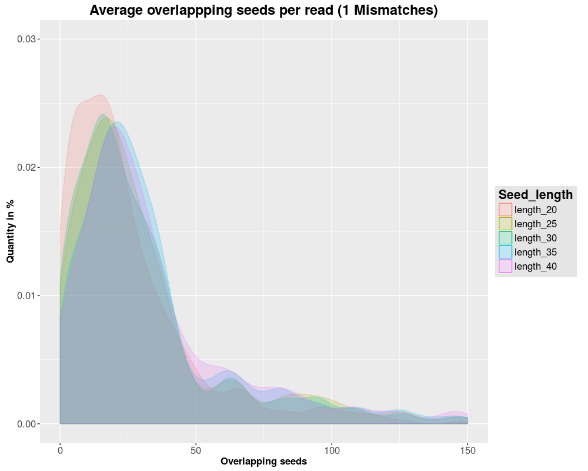
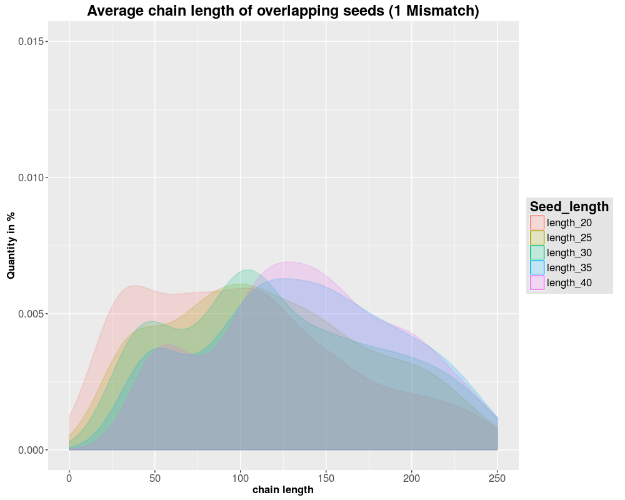


1. Analysis of the Bowtie 2 mapping for different seed lengths. All annotated seeds (red) are compared to the seeds with a valid mapping (blue) and to those that are not filtered out during the chaining step (green). One block of bars of this three colours represent the result for one seed length.

Next, the coverage of the reads was analysed, for both, the coverage caused by the mapped seeds and the coverage resulting after filtering and chaining those seeds. In this plots the lowest lines correspond to the highest average coverage of the reads. In both cases, the greatest length implicate the highest coverage, while shorter seed lengths resulting in a lower coverage. No great differences appear regarding the coverage before and after chaining, this shows that the seeds map well on the read and that the filtering and chaining did not eliminate the right seeds. This is correlated to the average chain length and to the average number of overlapping seeds pro read. Also, those two plots show that greater seed length implicate more overlapping seeds when forming a chain and consequentially longer chains, this is evidenced by the movement to the right of the density of the reads for longer lengths.

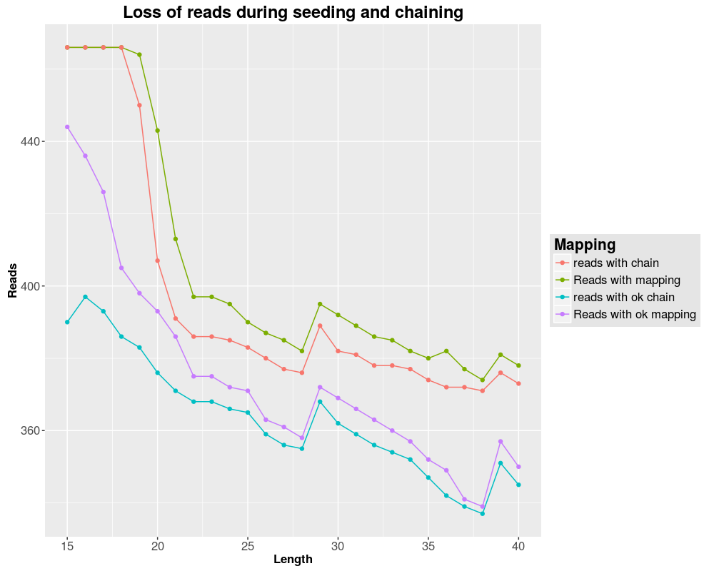


2 each line corresponds to the cumulative distribution of the average read coverage caused by the seeds (left) and chains (right) for a certain seed chain.

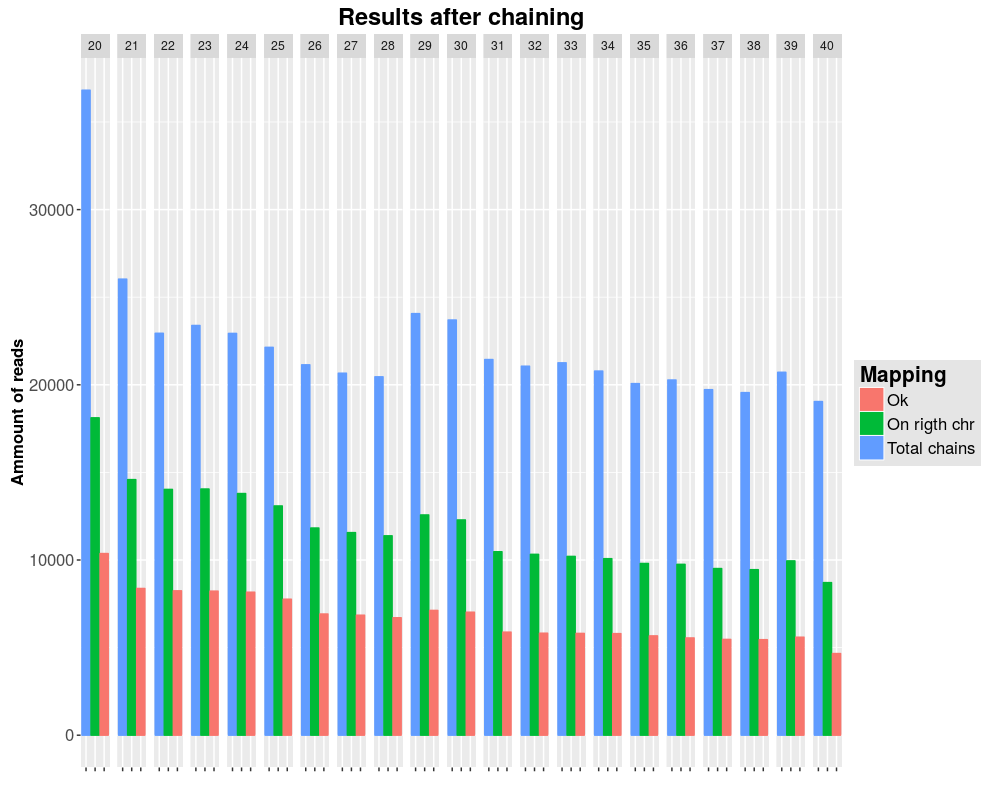


3 Two splots showing characteristic of the choosen seed length. The left one shows the density of the average chain length while the right one the average overlapping seeds for each read.

The next plot shows the loss of reads during the first filtering steps.



4 Each dot represents the amount of reads (y axis) after each filtering stage for a certain seed length (x axis). The highest line shows the reads for which at least one seed mapped on it (green) and the red line the reads with at least a chain on it. The purple and the blue line show the reads with at least, respectively one seed or one chain mapping on the right position in the genome.



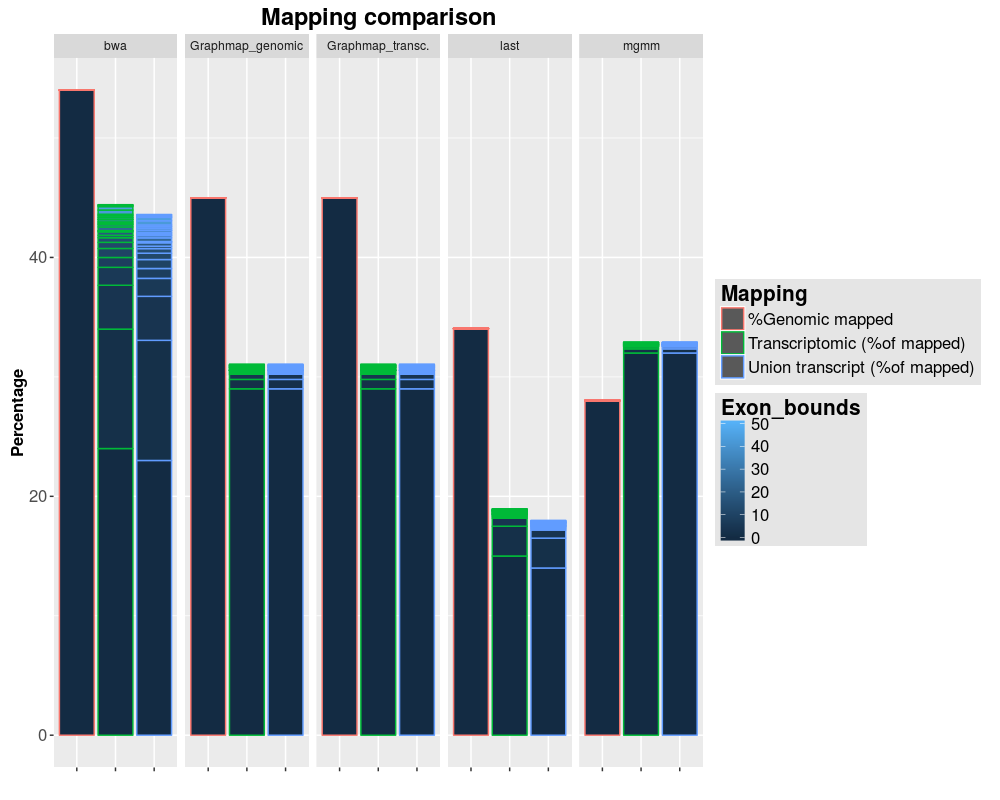
5 Analysis of the results after chaining the reads. The leftmost bar in a bar triple shows the total number of created chains over all reads. The second one shows number of chains mapping on the right chromosome of the read and the last one the chains intersecting the reads genomic regions.

# 6. Results

In this part a comparison of the created mapper to the present mapper is done. The first plot simply show the position of the mappings for the different mapper. The plot shows the percentage of reads mapping on genes over all mappings; and the percentage of transcriptomic mappings of the genomic mapped reads. Not only in this case but in all other analysis too, there are no differences between the normal GraphMap mapping and the mapping with the transcriptomic adaption (if adding a gtf file when calling it). The reason could be that this feature was published only two month ago and that the first version with this feature was used.

BWA has the highest percentage of genomic mapped reads but only allowing an exon bound tolerance of at least one base a good transcriptomic mapping is achieved. The significant differences using tolerance on the exon bounds is a consequence of no available information about the position of the introns and exons, since BWA is a genomic mapper and can not use a gtf file for the extraction of these information.

A characteristic attribute of LAST is the elevate number of annotated mappings. When analysing the mapping of our seeds, we saw that in most cases, a mapping of the seeds corresponds to an annotated alignment of LAST. This attribute is respected when analysing the bars of LAST. In fact, less than 40% of all mappings mapped on genes and only 20% of these are mappings on transcripts.



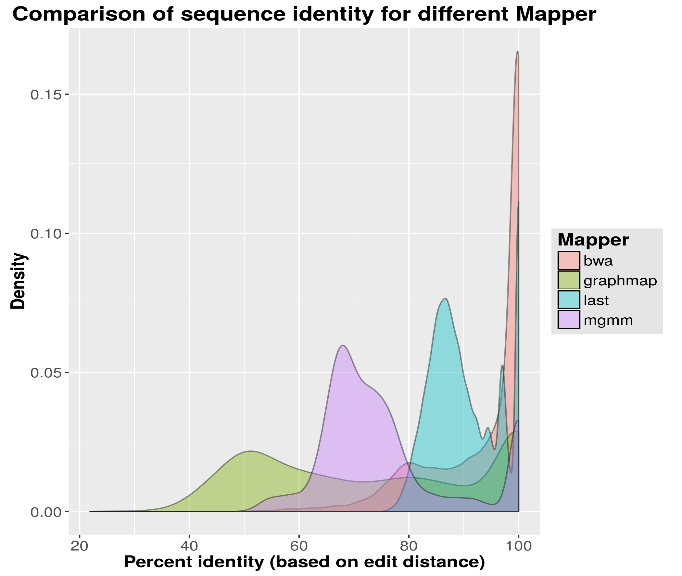
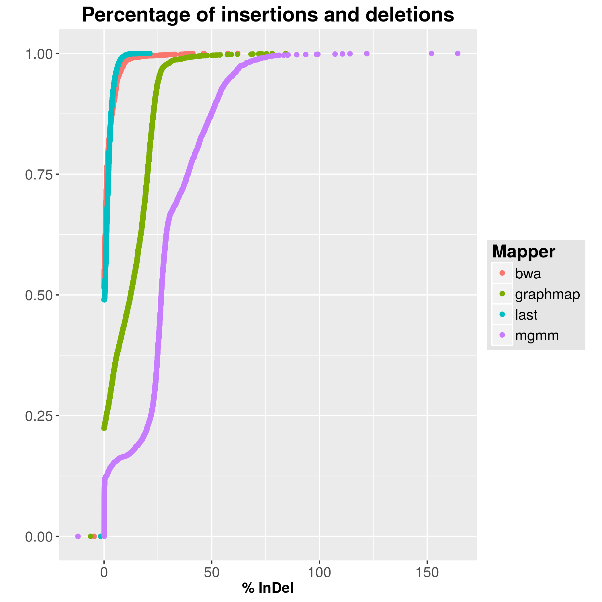
6 plot showing the three analysed mappers (BWA, GraphMap transcriptomic and genomic, LAST) in comparison to our MGMM mapper. A simulated dataset of 5538 reads was used for the plot. Each bar triple represent one mapper. The leftmost bars indicate the percentage of genomic mapped reads over all annotated alignments for that mapper. Then the green ones indicate the percentage of transcriptomic mapping reads of the genomic mappings (orange); and the last one (blue) the percentage of mappings of the genomic mapped reads on the union transcript. Tolerance from 1 to 50 bases on the exon bounds are shown with the stacks in the bars.

The analysis of sequence identity and InDel rate for the simulated data shows that BWA and LAST have the highest mapping quality among all four tested mappers. Both have mappings with a very low InDel rate (Fitting the 1% simulated InDel rate) while mapping a lot of reads with very low edit distances (As seen at the 100% peak in the percent identity plot).

LAST maps most reads with a sequence identity of ~85%, which can be explained with the large number of mappings the algorithm produces. LAST creates a mapping even for regions where only small regions actually map the read, resulting in a vast amount of false positives (With lower identity than the true positive matches).

The sequence identity of GraphMap is almost equally distributed with a small peak at 50% and 100% and in average lower than BWA or even LAST. This is probably connected to the fact that GraphMap doesn't seem to skip introns, resulting in very high edit distances for every read containing multiple exons.

MGMM has a higher sequence identity of GraphMap because of the pathing which reduces the maximum possible edit distance dramatically. The general behaviour looks similar to LAST with a thin peak at 100% and a wider peak at ~70% (Which correlates with the 30% mismatch rate of the simulated data). The InDel rate of MGMM on the other side is too large (With a median of ~25% instead of the simulated 1%), probably because of the fuzzy pathing and a too small gap penalty during the alignment.



On our way to reach a high-sensitive read-to-reference alignment, we had been able to compare the mappings of different mappers (GraphMap, BWA-MEM, Last, MGMM) on the simulated transcriptomic data set. The most mappings on the right position had BWA-MEM with 1108 of 6000 reads, then GraphMap with 1061 and Last with 897. This could be caused by the exon boundaries, because when looking on the quote of right mapped alignments divided by intersecting alignments, GraphMap mapped only 37% of partial alignments on the correct position, the others had all less than 9%. We missed our goal by mapping zero reads to the right position. The amount of mappings also differ strongly, our mapper MGMM and GraphMap had mostly primary SAMRecord, BWA-MEM made way too much secondary alignments (14594 SAMRecords all together) and Last failed in giving a usable output by having 159283 SAMRecords. By comparing the chromosome that had been chosen for the alignment, there are clear benefits of BWA-MEM and GraphMap that are aligning ca. 98% of the SAMRecord on the right chromosome, MGMM only could align 35,9% right and the worst had been Last with a quote of 20%.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | GraphMap | Last | BWA | MGMM |
| Annotated Alignments | 5538 | 159283 | 14594 | 5983 |
| Mapped Reads | 4023 | 159283 | 13698 | 5905 |
| Wrong Chromosome | 79 | 126809 | 191 | 3788 |
| Right Cromosome | 3944 | 32474 | 13507 | 2117 |
| Partial | 2861 | 17251 | 12314 | 1306 |
| Ok | 1061 | 897 | 1108 | 0 |
| Consistent | 1061 | 897 | 1108 | 0 |