Towards selective-alignment: producing accurate and sensitive alignments using quasi-mapping

Hirak Sarkar[1,†], Mohsen Zakeri[1,†], Laraib Malik[1], and Rob Patro[*1]

1 Department of Computer Science, Stony Brook University, Stony Brook, 11790
† These two authors contributed equally to the paper

Abstract

We introduce a novel algorithm for selectively aligning high-throughput sequencing reads to a transcriptome. This algorithm attempts to bridge the gap between fast "mapping" algorithms and more traditional alignment procedures. The former of these simply provide the transcripts, loci and orientations that likely generated a read (or, perhaps, simply the transcripts that are "compatible" with a read), while the latter produces the optimal edit distance and nucleotide-level correspondence between the read and reference sequences. We adopt a hybrid approach that is able to produce accurate alignments while still retaining much of the efficiency of fast mapping algorithms. To achieve this, we make fundamental modifications to an existing mapping algorithm, quasi-mapping, which increases the sensitivity of the procedure. Additionally, unlike the strategies adopted in most aligners which first align the ends of paired-end reads independently, we introduce a notion of co-mapping. This procedure exploits relevant information between the "hits" from the left and right ends of paired-end reads before full alignments or mappings for each are generated, which improves the efficiency of filtering likely-spurious alignments. Finally, we demonstrate the utility of selective alignment in improving the accuracy of efficient transcript-level quantification from RNA-seq reads. Specifically, we show that selective-alignment is able to resolve certain complex mapping scenarios that can confound existing fast mapping procedures, while simultaneously eliminating spurious alignments that fast mapping approaches can produce.

Availability and implementation: Selective-alignment is implemented in C++11 as a part of *RapMap*, and is available as open source software, under GPL v3, at `https://github.com/COMBINE-lab/RapMap/tree/selective-alignment`

Keywords and phrases Mapping, Alignment, RNA-seq, Quantification, Selective Alignment

1 Introduction

Since the introduction of high-throughput, short read sequencing technologies, many algorithms and tools have been designed to tackle the problem of aligning short sequenced reads to a reference genome or transcriptome accurately and efficiently. While there exist "full-sensitivity" aligners (e.g. RazerS3 [26], Masai [22]) which guarantee to find all reference positions within a given edit-distance threshold of a read sequence, the most widely-used tools employ heuristic strategies to enable much faster alignment of reads in the typical case (i.e., only a small number of easy-to-find candidate locations exist for each alignment). The common procedure followed by these tools for aligning reads can be divided into two major steps. The first is finding potential alignment locations for the read using a pre-processed index that is generated from the reference genome or transcriptome. Then, in the second step, the potential locations are filtered, and reads are aligned to the positions that pass the initial filtering, based on a variety of heuristics. The exact method for generating the

[*] corresponding author rob.patro@cs.stonybrook.edu

initial index varies for each tool. For example, tools like Bowtie [7], Bowtie2 ([6]), BWA [11], and BWA-mem [10] use Burrows-Wheeler transformation (BWT) based indices, whereas, k-mer based indices are used by tools such as Subread-aligner [13], Maq [12], SNAP [29], and GMAP and GSNAP [27]. Similarly, the heuristic for choosing the most probable locations is also different. However, each method is based on the principle of trying to find the reference loci that support the best (or near-best) alignment score between the read and the reference. Repeating this for a large number of reads comes with a considerable cost, in terms of computation. Some tools, like STAR [4], considerably speed up the alignment process by combining efficient heuristics with data structures (like the uncompressed suffix array) that trade working memory for exact pattern lookup speed. Recently, tools like HISAT [5] have also demonstrated that cache-friendly compressed indices (the hierarchical FM index in this case) can provide similarly efficient pattern search, even with a very moderate memory budget. The alignment of sequenced reads to the reference is the first step in pipelines leading to various downstream studies, such as estimation of transcript abundances and differential expression analysis, calculation of splicing rates [21, 25], and detection of fusion events [16, 3].

While alignment is a staple of many genomic analyses, it sometimes represents more information than is actually necessary to address the analysis at hand. For example, recent tools like *Sailfish* [18] (including its quasi-mapping-based variant [24]), *kallisto* [2], and *Salmon* [17], demonstrate that much of the information provided by aligners is unnecessary for accurate transcript quantification. By avoiding traditional alignment, these tools are much faster than their alignment-based counterparts. Furthermore, by building the mapping phase of the analysis directly into the quantification task, they dispense with the need to write, store, and read, large intermediate alignment files. However, these “mapping-based” tools, while highly-efficient, have the disadvantage of potentially losing sensitivity or specificity in certain adversarial cases where alignment-based methods would perform well. For example, in the presence of paralogous genes, with high sequence similarity, there is an increased probability that the mapping strategies employed by such tools, and the efficient heuristics upon which they rely, will mis-map reads between the paralogs (or return a more ambiguous set of mapping locations than an aligner, which expends effort to verify the returned alignments, would have) [1]. Similarly, in the case of *de novo* assemblies, poorly assembled contigs may have a larger number of mis-mapped reads due to lower sensitivity (here, the issue would be primarily due to aberrant exact matches masking the true origin of a read).

In this paper, we present a novel concept, selective-alignment, that extends the quasi-mapping algorithm to compute and store alignment information where necessary. The reads for alignment are chosen based on certain criteria calculated using mapping. This strikes a balance between speed and accuracy; not compromising the superior speed of fast mapping algorithms, while also addressing some of the challenges mentioned above. Specifically, the motivation for selective-alignment is to enhance both the sensitivity and specificity of fast mapping algorithms by reducing or eliminating cases where spurious exact matches mask true mapping locations as well as cases where small exact matches support otherwise poor alignments. We build our selective-alignment algorithm atop the framework of *RapMap* [24], which uses an index that combines a fixed prefix length hash table and an uncompressed suffix array [14]. We introduce a coverage-based consensus scheme to identify critical read candidates for which alignment is necessary. We explore challenging cases where the heuristics used by fast mapping algorithms fail to locate the correct locations for a read, but where traditional aligners do not, and show that selective-alignment enables us to reach the accuracy comparable to an aligner in *considerably* less time. We also introduce filtering steps based

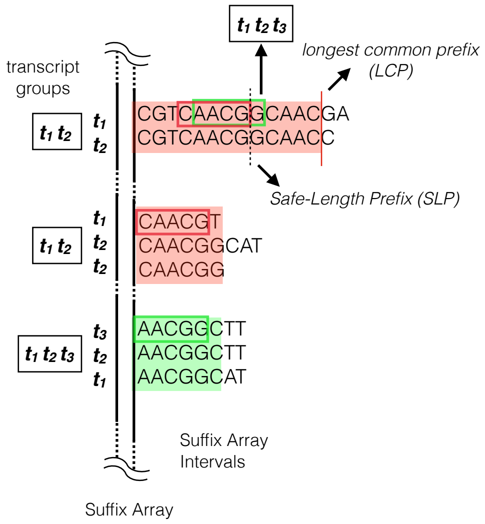

Figure 1 Calculation of safe length from the suffix array data structure. The transcripts present in each suffix array interval determine the relevant transcript sets, and which k-mers will be considered as intruders. Detection of a k-mer that maps to suffix array interval labeled (t_1, t_2, t_3) determines the safe-length here.

on edit distance to further refine probable alignments in order to enhance quantification estimates (e.g., eliminating situations where the best mapping is still unlikely to represent the true origin of the read).

2 Methods

The process of selective-alignment builds upon many of the basic data structures and ideas of quasi-mapping; yet there are a number of fundamental distinctions. Hence, we begin with a brief summary of the data structures backing the quasi-mapping implementation of *RapMap*. To start with, the index built on the transcriptome in selective-alignment is a combination of a suffix array and a hash table constructed from unique k-mers and suffix array intervals. Formally, given a suffix array, $\texttt{SA}[T]$, constructed from the transcriptome sequence, T, we construct a hash table, h, that maps each k-mer, k_i, to an interval, $\texttt{I}\,(k_i) = [b, e)$, if and only if all the suffixes within interval $[b, e)$ contain the k-mer k_i as a prefix. In addition to suffix array intervals, we also store two extra pieces of information for each interval; the longest common prefix (LCP) and the k-safe-LCP corresponding to the interval. These are detailed below.

2.1 Safe-length

Here, we formally define the concept of safe-lengths in terms of k-safe-LCPs. The determination of k-safe-LCPs starts by labeling each suffix array interval with the length of its corresponding longest common prefix and the associated transcript set it represents. Formally, $|\texttt{LCP}\,(\texttt{T}\,[\texttt{SA}\,[b]]\,, \texttt{T}\,[\texttt{SA}\,[e-1]]))\,|$ for an interval $[b, e)$ is the length of the common prefix of the suffix starting at $\texttt{T}\,[\texttt{SA}\,[b]]$ and that starting at $\texttt{T}\,[\texttt{SA}\,[e-1]]$.

Given a k-mer k_i and the related interval $\texttt{I}\,(k_i) = [b, e)$, for all $p \in [b, e)$, we consider each transcript t_i such that $\texttt{SA}\,[p]$ occurs in transcript t_i in the concatenated text T. Then, we can construct for this interval a set, $\mathcal{C}^i = \{t_{i1}, t_{i2}, \dots\}$, which denotes the set of distinct transcripts that appear in this suffix array interval. We note that this notion discards duplicate appearances of the same transcript in this interval. However, it is also possible to define k-safe-LCPs in a manner that requires relative transcript positions to be consistent as well, though we don't adopt such a definition here. This defines an equivalence relation over suffix intervals such that two intervals $\mathtt{I}\,(k_i)$ and $\mathtt{I}\,(k_j)$ are equivalent if and only if $\mathcal{C}^i = \mathcal{C}^j$.

Given a suffix array interval $\mathtt{I}\,(k_i) = [b, e)$, we check, sequentially, each of the k-mers in the suffix $\mathtt{T}\,[\mathtt{SA}\,[b, :]]$. We define the k-safe-LCP for $\mathtt{I}\,(k_i)$ to end if any of the following conditions is encountered: (1) we reach the end of the LCP of this interval, (2) we encounter a k-mer k_j such that $\mathcal{C}^j \nsubseteq \mathcal{C}^i$ or (3) we encounter a k-mer k_j such that the reverse complement of k_j appears elsewhere in the transcriptome. When we encounter case (2) or (3), we call the k-mer k_j an *intruder*. That is, this k-mer will potentially alter our belief about the set of potential transcripts to which a sequence containing this k-mer maps (by strictly expanding this set), or the orientation with which it maps to the transcriptome. The safe length and the corresponding prefix are denoted as k-safe-LCPs, and we denote the k-safe-LCP of a particular interval $\mathtt{I}\,(k_i)$ as a safe-length prefix or $\mathtt{SLP}(k_i)$.

As shown in Figure 1, the safe length determination for the top suffix array interval starts with matching k-mers within the longest common prefix. The k-mer "CAACG" is the last k-mer that maps to a suffix array interval labeled with (t_1, t_2). The next k-mer "AACGG", on the other hand, maps to a suffix array interval (shaded in green) labeled with (t_1, t_2, t_3), thereby determining the safe-length, shown as a dotted line.

For each k-mer in the hash table, we store the length of the LCP and k-safe-LCP along with the corresponding suffix array interval.

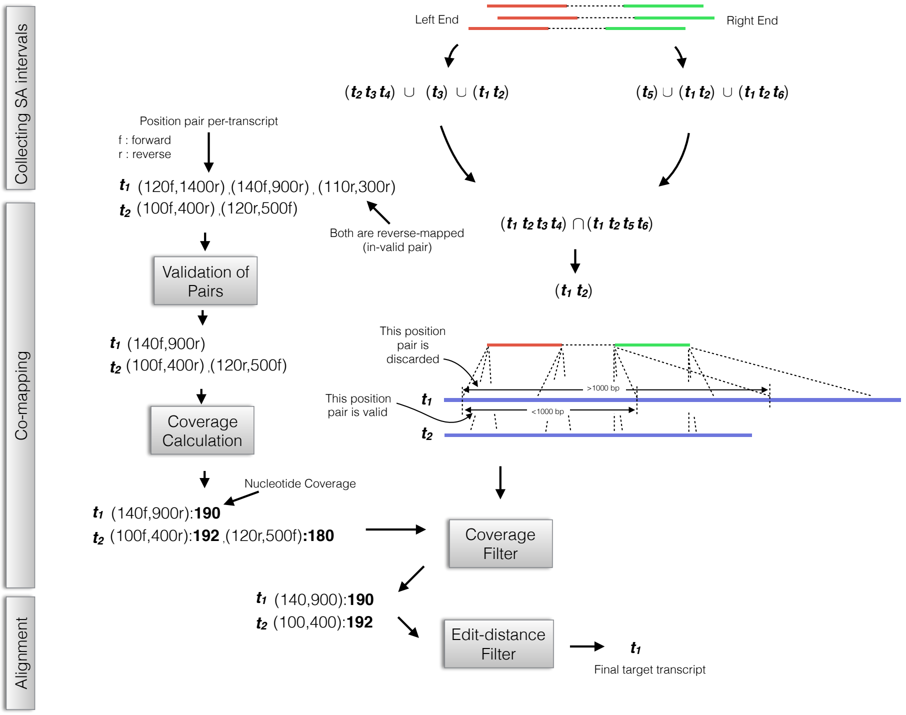

Figure 2 The three main steps of the selective-alignment process are demonstrated here. First, suffix array "hits" are collected. Then, in co-mapping, spurious mappings are removed by the orientation filter and then distance filter. At most a single locus per-transcript is selected based on the coverage filter. Finally, an edit-distance-based filter is used to select the valid target transcripts.

2.2 Selective alignment of read sequence

2.2.1 Discovering relevant suffix array intervals

As shown in Figure 2, the selective-alignment approach can be broken into three steps: collecting suffix array intervals, co-mapping, and selective alignment. Getting the suffix array intervals for a query read closely follows the quasi-mapping approach. Similar to quasi-mapping, it involves iterating over the read from left to right and repeating two steps. First, hashing k-mer from the read sequence and then discovering the corresponding suffix array intervals. The process of k-mer lookup is aided by the safe-length stored in the index (discussed in Section 2.1). We make use of the inbuilt lexicographic ordering of the suffixes in the suffix array by skipping the required k-mer lookup whenever possible. Given a matching k-mer, k_i, from the read sequence, we extend the match and find the longest substring of the read that matches within $\texttt{SLP}(\texttt{I}\,(k_i))$. The matched substring can be regarded as maximal mappable prefix (MMP) [4], that resides within the established safe length. We call this a maximal mappable safe prefix (MMSP — eliding k where implied). For a k-mer, k_i, and interval, $[b, e)$, we note that $|\texttt{SLP}\,(\texttt{T}\,[\texttt{SA}\,[b]]\,, \texttt{T}\,[\texttt{SA}\,[e-1]])\,| \geq \ell_{\text{MMSP}i}$, where $\ell_{\text{MMSP}i}$ is the length of MMSPi, the MMSP starting at position i in the read. The next k-mer lookup starts from (MMSP$i - k + 1$)-th position. For a given k-safe MMP, MMSPi, all k-mers, k_j, occurring in the prefix satisfy $\mathcal{C}^j \subseteq \mathcal{C}^i$. Given the construction mechanism described above we have the following theorem:

▶ **Theorem 1.** *Querying only k-mers at the beginning and end of each k-safe MMP yields the same set of transcripts as querying every k-mer on the read.*

Proof. Assume that, given a read sequence r, we have skipped m times to navigate through all k-mers within the read sequence. The m k-mer lookups encountered in the process are denoted as $\{k_{m_1}, \ldots, k_{m_j}\}$. Without loss of generality, we must only show that for a given $k_{m_i} \in \{k_{m_1}, \ldots, k_{m_j}\}$, there is no k-mer $k_{m_{i'}} \in r[m_{i-1} : m_i]$ appearing in a transcript, t, for which we have not considered a suffix array interval encoding t.

We can prove this claim by contradiction. Assume there exists such a k-mer, $k_{m_{i'}} \in r[m_{i-1} : m_i]$, for which we miss at least one correct transcript. According to the definition given in Section 2.1, we know that there exists a transcript set $\mathcal{C}^{m_{i'}}$ corresponding to k-mer $k_{m_{i'}}$. Following the definition of the corresponding transcript sets, the claim further reduces to existence of a transcript $t_x \in \mathcal{C}^{m_{i'}}$ such that $t_x \notin \mathcal{C}^{m_{i-1}}$, so $\mathcal{C}^{m_{i'}} \not\subseteq \mathcal{C}^{m_{i-1}}$. This directly contradicts the definition of a maximum mappable safe prefix. In other words, $\forall m_{i'}$ such that $m_{i-1} \leq m_{i'} < m_i$, $\mathcal{C}^{m_{i'}} \subseteq \mathcal{C}^{m_{i-1}}$. Therefore, no such transcript t_x can exist. ◀

Given all the suffix array intervals collected for a read end (i.e. one end of a paired-end read), we take the *union* of all the transcripts they encode. Formally, if a read r maps to suffix array intervals labeled with $\mathcal{C}^{r_1}, \ldots, \mathcal{C}^{r_n}$, then we consider all transcripts in the set $\{\mathcal{C}^{r_1} \cup \mathcal{C}^{r_2} \cup \ldots \cup \mathcal{C}^{r_n}\}$, and the associated positions implied by the suffix array intervals. As shown in Figure 2; this step is done before co-mapping. We note that, in practice, we actually adopt a hybrid approach to collecting the suffix array intervals. Specifically, when the k-safe-LCP only has a length of k, instead of moving to the next k-mer, we jump by $|r|\,/10$ nucleotides (where $|r|$ is the read length) before looking up the next k-mer, otherwise (if the k-safe-LCP is $\geq k$), we skip by the MMSP length as described. This prevents us from performing excessive lookups in low-complexity and repetitive regions of the transcriptome. We observe that, in practice, the k-safe-LCP, and hence the MMSP lengths tend to be large (Figure 3).

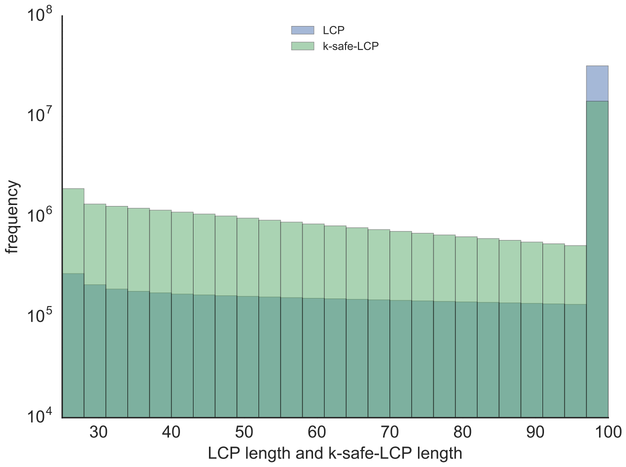

Figure 3 The distribution of k-safe-LCP lengths and LCP lengths are similar and tend to be large in practice (human transcriptome). Here, we truncate all lengths to a maximum value of 100 (so that any LCP or k-safe-LCP longer than 100 nucleotides is placed in the length 100 bin).

2.3 Co-Mapping

After collecting the suffix array intervals corresponding to left and right ends of the read, we wish to exploit the paired-end information in determining which potential mapping locations might be valid. Hence, from this step onwards we use the joint information for determining the position and target transcripts. Given the suffix array intervals for individual ends of a paired end read, the problem of aligning both ends of the pair poses a few challenges. First, a single read can map to multiple transcripts, and we wish to report all equally-best loci. Second, there can be multiple hits on a single transcript (e.g., if a transcript contains repetitive sequence), and extra care must be taken to determine the correct mapping location. Finally, there may be hits that do no yield high-quality alignments (i.e. long exact matches that are nonetheless spurious). To address the first and third points, we employ an edit distance filter to discard spurious and sub-optimal alignments. To address the second challenge, we devise a consensus strategy to choose at most one unique position from each transcript.

Before applying the above mentioned strategy, we remove transcripts that do not contain hits from both the left and right ends of the read. Formally, given two ends of a read r as, r^{e_1} and r^{e_2}, and the corresponding suffix array intervals labeled with $\mathcal{C}^{r_1^{e1}}, \ldots, \mathcal{C}^{r_n^{e1}}$ and $\mathcal{C}^{r_1^{e2}}, \ldots, \mathcal{C}^{r_m^{e2}}$ respectively, we only consider transcripts present in the set $(\mathcal{C}^{r_1^{e1}} \cup \ldots \cup \mathcal{C}^{r_n^{e1}}) \cap (\mathcal{C}^{r_1^{e2}} \cup \ldots \cup \mathcal{C}^{r_m^{e2}})$. We further refine this set by checking the validity of the alignments these hits might support. Currently, we use two validity checks illustrated in Figure 2. First, we apply an orientation-based check, and second we employ a distance-based check. The orientation check removes potential mappings which have an orientation inconsistent with the underlying sequencing library type (e.g., both ends of a read mapping in the same orientation). The distance-based check removes potential alignments where the implied distance between the read ends is larger than a given, user-defined threshold.

2.3.1 Coverage based consensus

In selective-alignment, the potential positions on a transcript are scored by their individual coverage on the target transcript. *RapMap* [24] used a simplistic approach of choosing the first available position irrespective of the coverage profile. We observed that such a scheme

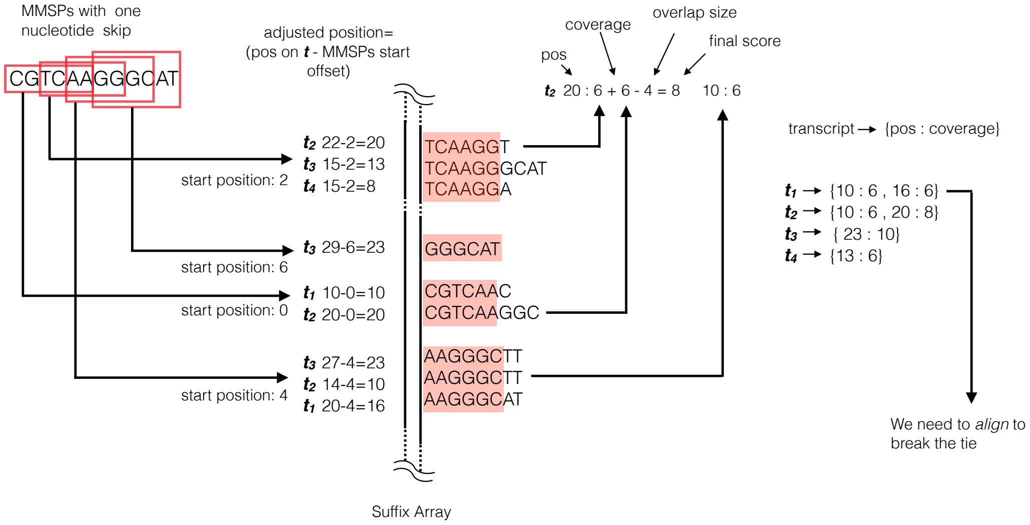

Figure 4 The MMSPs corresponding to a read, are derived from multiple suffix array intervals. Here, all MMSPs happen to be of length k. The coverage scheme finds out the exact positions on each transcript by adjusting the starting position of the matches. The total score takes into account the positions where matches overlap. A position is chosen by selecting the locus with maximum coverage.

can sometimes lead to selecting positions that support a suboptimal alignment. One such situation is depicted in Figure 4. The coverage mechanism employed in selective-alignment makes use of the MMSP lengths collected during a prior step of the algorithm rather than simply counting k-mers.

2.3.2 Selecting the best candidate transcript

Once the positional ambiguity within a transcript is resolved, the next step is selecting the best candidate transcripts from a set of mappings. Since mapping relies on finding exact matches, the length of the matched subsequence between the read and reference can sometimes be misguiding when comparing different candidate transcripts. That is, the transcripts with the longest exact matches do not always support optimal alignments for a read. At this point in our procedure, we follow the approach taken by many conventional aligners, and use an existing optimal alignment algorithm to compute the edit distance, by which we select the best candidate transcripts.

When performing alignment, we assume that a given read aligns starting at the position computed in the previous steps. This helps us to reduce the search space within the transcript where we must consider aligning the read, and thereby considerably reduces the cost of alignment. To align the read at a specific position on the transcript and calculate the edit distance between them, we use $Myer's$ bounded edit distance bit-vector algorithm [15], as implemented in `edlib` [23]. For a fixed maximum allowable edit distance, this algorithm is linear in the length of the read. We note that the bounded edit distance algorithm we employ will automatically terminate an alignment when the required edit distance bound is not achievable.

We remove all alignments with edit distance greater than a user-provided threshold. This is similar to the approach used by many existing aligners, and allows us to specify that even the best alignment for a given read may have too many edits to believe that it reasonably originated from a known transcript in the index. An appropriate threshold should be based on the expected error rate of the instrument generating the sequenced reads, and a very low threshold can lead to decreased mapping rate.

2.3.3 Enhancement of quantification accuracy based on edit distance score

We also investigated the effect of incorporating edit distance in downstream quantification. Since we integrated the selective-alignment scheme into the quantification tool *Salmon* [17], the edit distance scores from selective-alignment can be used as a parameter to *Salmon*'s inference algorithm.

In the framework of abundance estimation, we define the conditional probability of a fragment, f_j, originating from a transcript, t_i, as $P(f_j \mid t_i)$. Given the edit distance between the fragment and the transcript, we can incorporate this parameter into this conditional probability. *Soft* filtering introduces a term in the probability based on the sum of the edit distances of the read ends for each fragment, $d_{i,j}$. We set this probability according to an exponential function, $P(e_{i,j}|f_j, t_i) = e^{-4d_{i,j}}$. The aggregate of threshold filtering and *soft* filtering can be described as follows:

$$\Pr\left(f_j \mid d_{i,j}, t_i\right) = \begin{cases} 0 & d_{i,j} > \mathit{threshold} \\ e^{-4d_{i,j}} & d_{i,j} \leq \mathit{threshold} \end{cases}. \tag{1}$$

The other approach for further refining the set of candidate mappings is *strict* filtering. Using this approach, only the hit(s) that have the minimum edit distance in the set of mappings are stored and the others are discarded. For example, given a fragment f_1 that maps to transcripts t_1, t_2, t_3 with edit distances $e_{1,1} = 10$, $e_{1,2} = 12$ and $e_{1,3} = 10$, *strict* filtering will discard t_2. Hence, only the mappings to t_1 and t_3 will be reported for f_1. We can illustrate the aggregate of threshold filtering and *strict* filtering in the following equation:

$$\Pr\left(f_j \mid d_{i,j}, t_i\right) = \begin{cases} 0 & d_{i,j} > \min\limits_{t_k \in \Omega(r_j)} d_{k,j} \\ 1.0 & d_{i,j} \leq \min\limits_{t_k \in \Omega(r_j)} d_{k,j} \end{cases}, \tag{2}$$

where $\Omega(r_j)$ is the set of all transcripts to which read r_j maps.

For all experiments involving selective-alignment in this manuscript, we use *soft* filtering.

2.4 Shared LCPs prevents redundant alignments

Exploiting the common subsequences in the transcriptome is instrumental to the superior speed of fast mapping tools. Reads generated from exonic sequences common to multiple transcripts from the same gene or paralogous genes are the main source of ambiguous mapping. As we rely on the suffix array data structure to obtain the initial set of transcripts to which a read maps, there are cases where exactly identical reference sequences all act as mapping targets for the read. For a suffix array interval $[b, e)$, we identify such common subsequences by examining the *longest common prefix* (LCP) of the interval. If the length of the LCP is equal or greater than the length of the read, then the actual alignment to the underlying reference at these positions will be identical.

Given the computationally intensive nature of alignment, this approach can be exploited to avoid the process altogether for some set of reference positions by simply reusing the alignment information from one read transcript pair and then passing it to other transcripts that share the LCP. As a proof of concept, we profiled the specific cases where such redundant alignments have been skipped in our algorithm. We observed (Table 4) that for almost half of the read-transcript pairs, the alignment process can be avoided. Note that if the read sequence shares a complete match with the common prefix, meaning that maximum mappable prefix length (MMP length [24]) is equal to read length (i.e., the read matches the reference exactly at some set of positions), we can bypass the Meyer's edit distance algorithm call completely.

3 Results

To evaluate the effectiveness of selective-alignment, we coupled it with the state-of-the-art quantification tool *Salmon*. This enables us to measure the effect of different mapping/alignment algorithms on transcript-level quantification results directly, holding the statistical estimation procedure fixed. We also include *kallisto* in our benchmarks, which provides a perspective on pseudoalignment-based quantification. We measure the Spearman correlation and Mean Absolute Relative Differences (MARD) of read counts as performance metrics when comparing the different methods. We adopt the MARD definition from [17].

3.1 Adversarial Synthetic Data

Genes with multiple isoforms are among the most challenging cases for aligning/mapping reads, since isoforms of the same gene share exonic sequences and are prone to a high degree of multi-mapping. Particularly complex regions of the transcriptome can pose a challenge to fast mapping algorithms, since many exact matches may occur at loci other than those which generate an optimal alignment. This can cause spurious mappings to mask true alignment locations, harming both sensitivity and specificity. Here, we generate an adversarial synthetic dataset which highlights potential mis-mapping problems. We restrict both the generation and assessment to multi-isoform genes. From the set of all multi-isoform genes in the human transcriptome (referred to as ground set), we selected a subset of transcript isoforms from which to generate reads. Through this mechanism, we ensure that only a fraction of the ground set of transcripts are truly expressed. Since the unexpressed transcripts share considerable sequence with the expressed transcripts, we expect a high rate of ambiguous multi-mapping.

The simulation procedure is randomized, and can be described as a two-step process. In the first step, we select a set of target transcripts (the foreground set) and quantify their abundances using reads from an experimental RNA-seq sample. In the second step, we generate synthetic reads from this set of estimated abundances and quantify the resulting data using the entire background set.

To select the foreground set, we first examine each multi-isoform gene in the background set, and select it with probability p. Then, given this gene, we look at each isoform in turn and select it with probability q. Therefore, the number of truly expressed transcripts never exceeds $100 \times pq$ percent of the number of transcripts in the ground set. For the two simulated datasets used here, $100 \times pq$ is 30 and 60, respectively ($p = 0.6, q = 0.5$ and $p = 1.0, q = 0.6$). The motivation for this experimental set up comes from a previous analysis of the effect of different quantification procedures on expression "bleed through"[4].

[4] https://cgatoxford.wordpress.com/2016/08/17/why-you-should-stop-using-featurecounts-htseq-or-

Table 1 Performance of methods in terms of quantification accuracy on two foreground sets, 30% and 60%. quasi-mapping is the mapping approach used by *RapMap*, all alignments/mappings are quantified with *Salmon* or *kallisto*.

Method	Foreground set	Spearman	MARD	time (s)
HISAT 2	30%	0.849	0.077	1,180
kallisto	30%	0.856	0.075	25
quasi-mapping	30%	0.857	0.074	48
selective-alignment	30%	0.907	0.044	92
Bowtie 2	30%	0.911	0.042	1,344
HISAT 2	60%	0.860	0.137	1,451
kallisto	60%	0.876	0.128	28
quasi-mapping	60%	0.875	0.127	48
selective-alignment	60%	0.904	0.096	90
Bowtie 2	60%	0.906	0.094	1,351

To simulate data, *RSEM* [9] was run on sample `N12716_7` of the Geuvadis study [8], with the selected foreground set of transcripts (30% and 60% respectively) used as a reference to learn the model parameters and estimate true expression. The learned model is then used to generate 15 million, 75bp paired-end reads. These reads generated from this foreground set are then aligned/mapped to the ground set (i.e., all multi-isoform transcripts taken from protein coding transcripts of GRCh38.p10) using *Bowtie 2*, *HISAT 2*, *RapMap*, *kallisto* and selective-alignment. Subsequently, transcripts are quantified by *Salmon* using the relevant alignments/mappings as input (except in the case of *kallisto*). The alignment mode of *Salmon* enables us to use *HISAT 2* and *Bowtie 2* output as a direct input to the quantification module — thereby reducing variability due to differences in the underlying model. To achieve the most sensitive alignment, *Bowtie 2* and *HISAT 2* are run with the *Bowtie 2* alignment options used by *RSEM*. When processing alignments, *Salmon* was run with `--useRangeClusterEqClasses` [30] and `--useErrorModel`. With selective-alignment, *Salmon* was run using `--useRangeClusterEqClasses`, `--softFilter` (discussed in Section 2.3.3) and an edit distance threshold of 7. *kallisto* was run with default parameters. Both the *Salmon* and *kallisto* indices were built with $k = 25$.

As displayed in Table 1, for the dataset where at most 30% of transcripts are truly expressed, the *Bowtie 2*-based (and selective-alignment-based) methods perform better than the fast mapping approaches. We note that *HISAT 2*, presumably, is not optimized for mapping directly to the transcriptome (i.e., it is developed primarily as a genome-based spliced-read aligner), and a different set of parameters might lead to a more accurate result.

In the experiment where at most 60% of transcripts are truly expressed, the accuracy of all methods begins to converge. Though we have designed these experiments to be adversarial in nature, they nonetheless raise an interesting point about how divergence between the true set of expressed transcripts and those considered during quantification might affect accuracy. Specifically, aligning/mapping against a larger and more comprehensive set of potential isoforms need not always yield superior results. When unexpressed isoforms share considerable sequence with those that are truly expressed, the probability of mis-assigning ambiguously mapping reads can increase. Though this is true regardless of how reads are

cufflinks2-and-start-using-kallisto-salmon-or-sailfish/

Table 2 Quantification results with different methods for aligning/mapping reads on transcriptome wide synthetic data. All quantifications are computed with *Salmon* or *kallisto*.

Method	Spearman	MARD	time (s)
HISAT 2	0.788	0.242	2,353
kallisto	0.808	0.231	96
quasi-mapping	0.813	0.227	107
selective-alignment	0.823	0.215	202
Bowtie 2	0.825	0.214	2,860

aligned/mapped, alignment-based methods (and selective-alignment) seem less prone to mis-assignment in such cases.

3.2 Synthetic reads from human transcriptome

We have also explored the performance of different alignment-based and non-alignment-based methods on the full human transcriptome. We follow the procedure described in [2] to generate 30M, 75bp paired end reads using the *RSEM* simulator. Reads are mapped/aligned to the human transcriptome (Ensembl release 80 [28]) with different methods, and then quantified by *Salmon* (or *kallisto*). The Spearman correlation and MARD values for different methods in Table 2 demonstrates that the performance of both alignment-based and non-alignment-based methods are similar to each other at the transcriptome-wide scale (and when not focusing on adversarial situations). *Bowtie 2*-based quantification seems to marginally outperform the mapping-based methods. Selective-alignment's accuracy is very similar to that of *Bowtie 2*, but it requires considerably less time (it is similar to the fast mapping-based methods in this respect).

Transcriptome-wide assessments on synthetic data, like that explored in this experiment, suggest that fast mapping-based methods generally perform well (and similar to alignment-based methods). However, small global differences in quantification accuracy at the transcriptome-wide scale tend to arise from larger differences in the quantification of particular transcripts (e.g., those where accurate mapping tends to be difficult, and where additional modeling fidelity is required to obtain accurate estimates [30]). Such differences also arise, and tend to be somewhat larger, when analyzing experimentally-derived data, as we do in Section 3.3.

3.3 Experimental reads from human transcriptome

We have also benchmarked our proposed method selective-alignment method, on experimental data from SEQC(MAQC-III) consortium [20] (NCBI GEO accession `SRR1215996` - `SRR1217000`). Each of five technical replicates consists of $\sim$11M, 100bp, paired-end reads, sequenced on an Illumina Hiseq 2000 platform. The options used for all methods are the same as those mentioned in Section 3.1. In Table 3, we compare the quantification results produced by different methods. Here, we note that we do not know the ground truth, and so we instead measure the overall concordance between different approaches. Each individual cell contains the average obtained across all five samples. High Spearman correlation and low MARD value between *Bowtie 2* and selective-alignment show that selective-alignment produces results more similar to *Bowtie 2* than to the non-alignment-based methods.

Table 4 demonstrates how extension by the maximum mappable prefix, and also knowledge

Table 3 The Spearman correlation and MARDS between transcript abundances computed by all methods on experimental data. Each number is the mean on 5 different samples; the numbers in the lower left triangle of the matrix are the Spearman correlations and the ones in upper right are the MARD values.

Method	*HISAT 2*	*kallisto*	quasi-mapping	selective-alignment	*Bowtie 2*
HISAT 2	1 0	0.173	0.173	**0.087**	0.082
kallisto	0.868	1 0	0.018	0.137	0.152
quasi-mapping	0.870	0.990	1 0	0.137	0.151
selective-alignment	**0.932**	0.900	0.901	1 0	**0.057**
Bowtie 2	0.937	0.886	0.889	**0.958**	1 0

Table 4 The percentage of hits that skip the full alignment process due to extension by the maximum mappable prefix (MMP), or projection of duplicate alignments given the longest common prefix (LCP) sequences.

Sample	MMP skip	LCP skip
SRR1215996	44.661%	5.788%
SRR1215997	46.986%	7.854%
SRR1215998	40.614%	7.305%
SRR1215999	41.182%	6.884%
SRR1216000	45.321%	5.485%

of the longest common prefixes from the *RapMap* index, help to avoid performing independent alignment calls a considerable fraction of the time. The numbers in Table 4 show the percentage of hits for which no alignment is needed to obtain the edit distance values. For about half of the hits in each sample, we can skip alignment by considering the MMP and LCP information.

4 Conclusion

Recently, fast mapping approaches such as psuedoalignment [2] and quasi-mapping [24] have been developed for mapping RNA-seq reads to transcriptomes. Rather than generating full alignments, these approaches compute “mapping” information that is often sufficient for a number of given analysis tasks (e.g., transcript quantification [17, 2] or metagenomic abundance estimation [19]). Yet, there exist scenarios where such mapping approaches can go awry; either failing, by the greedy nature of their procedures, to find the true target of origin of a read, or by allowing spurious mappings to targets supported by exact matches that would nonetheless fail reasonable alignment scoring filters. Moreover, it is sometimes desirable to be able to produce, on demand, the edit distance or alignment that would result from a given mapping location. In this paper, we introduce a selective alignment algorithm that attempts to bridge the gap between these fast mapping algorithms and more traditional alignment algorithms. Selective alignment improves upon both the sensitivity and specificity of these mapping algorithms while making very moderate concessions with respect to the computational budget. To achieve this level of efficiency, a number of algorithmic innovations were required, some of which may be of general interest. In the future, we hope to expand upon the notion of selective alignment even further, both by improving the algorithm and implementation, and by exploring use cases where selective alignment applies. Such situations are those where fast mapping approaches are inappropriate and traditional alignment approaches are too slow.

5 Acknowledgements

This work has been supported by the National Science Foundation (BBSRC-NSF/BIO-156491).

References

1 Michael J Axtell. Butter: High-precision genomic alignment of small RNA-seq data. *bioRxiv*, page 007427, 2014.

2 Nicolas L Bray, Harold Pimentel, Páll Melsted, and Lior Pachter. Near-optimal probabilistic RNA-seq quantification. *Nature Biotechnology*, 34(5):525–527, 2016.

3 Nadia M Davidson, Ian J Majewski, and Alicia Oshlack. JAFFA: High sensitivity transcriptome-focused fusion gene detection. *Genome medicine*, 7(1):43, 2015.

4 Alexander Dobin, Carrie A Davis, Felix Schlesinger, Jorg Drenkow, Chris Zaleski, Sonali Jha, Philippe Batut, Mark Chaisson, and Thomas R Gingeras. Star: ultrafast universal rna-seq aligner. *Bioinformatics*, 29(1):15–21, 2013.

5 Daehwan Kim, Ben Langmead, and Steven L Salzberg. HISAT: a fast spliced aligner with low memory requirements. *Nature methods*, 12(4):357–360, 2015.

6 Ben Langmead and Steven L Salzberg. Fast gapped-read alignment with Bowtie 2. *Nature methods*, 9(4):357–359, 2012.

7 Ben Langmead, Cole Trapnell, Mihai Pop, Steven L Salzberg, et al. Ultrafast and memory-efficient alignment of short DNA sequences to the human genome. *Genome Biology*, 10(3):R25, 2009.

8 Tuuli Lappalainen, Michael Sammeth, Marc R Friedländer, Peter AC‘t Hoen, Jean Monlong, Manuel A Rivas, Mar Gonzàlez-Porta, Natalja Kurbatova, Thasso Griebel, Pedro G Ferreira, et al. Transcriptome and genome sequencing uncovers functional variation in humans. *Nature*, 2013.

9 Bo Li and Colin N Dewey. RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome. *BMC bioinformatics*, 12(1):323, 2011.

10 Heng Li. Aligning sequence reads, clone sequences and assembly contigs with bwa-mem, 2013.

11 Heng Li and Richard Durbin. Fast and accurate short read alignment with Burrows–Wheeler transform. *Bioinformatics*, 25(14):1754–1760, 2009.

12 Heng Li, Jue Ruan, and Richard Durbin. Mapping short dna sequencing reads and calling variants using mapping quality scores. *Genome research*, 18(11):1851–1858, 2008.

13 Yang Liao, Gordon K Smyth, and Wei Shi. The Subread aligner: fast, accurate and scalable read mapping by seed-and-vote. *Nucleic acids research*, 41(10):e108–e108, 2013.

14 Udi Manber and Gene Myers. Suffix arrays: a new method for on-line string searches. *siam Journal on Computing*, 22(5):935–948, 1993.

15 Gene Myers. A fast bit-vector algorithm for approximate string matching based on dynamic programming. *Journal of the ACM (JACM)*, 46(3):395–415, 1999.

16 Daniel Nicorici, Mihaela Satalan, Henrik Edgren, Sara Kangaspeska, Astrid Murumagi, Olli Kallioniemi, Sami Virtanen, and Olavi Kilkku. FusionCatcher-a tool for finding somatic fusion genes in paired-end RNA-sequencing data. *bioRxiv*, page 011650, 2014.

17 Rob Patro, Geet Duggal, Michael I Love, Rafael A Irizarry, and Carl Kingsford. Salmon provides fast and bias-aware quantification of transcript expression. *Nature Methods*, 2017.

18 Rob Patro, Stephen M Mount, and Carl Kingsford. Sailfish enables alignment-free isoform quantification from RNA-seq reads using lightweight algorithms. *Nature biotechnology*, 32(5):462–464, 2014.

19 L. Schaeffer, H. Pimentel, N. Bray, P. Melsted, and L. Pachter. Pseudoalignment for metagenomic read assignment. *Bioinformatics*, Feb 2017. doi:https://doi.org/10.1093/bioinformatics/btx106.

20 SEQC/MAQC-III Consortium and others. A comprehensive assessment of rna-seq accuracy, reproducibility and information content by the sequencing quality control consortium. *Nature Biotechnology*, 32(9):903–914, 2014.

21 Shihao Shen, Juw Won Park, Zhi-xiang Lu, Lan Lin, Michael D Henry, Ying Nian Wu, Qing Zhou, and Yi Xing. rMATS: robust and flexible detection of differential alternative splicing from replicate RNA-Seq data. *Proceedings of the National Academy of Sciences*, 111(51):E5593–E5601, 2014.

22 Enrico Siragusa, David Weese, and Knut Reinert. Fast and accurate read mapping with approximate seeds and multiple backtracking. *Nucleic acids research*, 41(7):e78–e78, 2013.

23 Martin Šošić and Mile Šikić. Edlib: a C/C++ library for fast, exact sequence alignment using edit distance. *Bioinformatics*, 33(9):1394, 2017.

24 Avi Srivastava, Hirak Sarkar, Nitish Gupta, and Rob Patro. RapMap: a rapid, sensitive and accurate tool for mapping RNA-seq reads to transcriptomes. *Bioinformatics*, 32(12):i192–i200, 2016.

25 Jorge Vaquero-Garcia, Alejandro Barrera, Matthew R Gazzara, Juan Gonzalez-Vallinas, Nicholas F Lahens, John B Hogenesch, Kristen W Lynch, and Yoseph Barash. A new view of transcriptome complexity and regulation through the lens of local splicing variations. *Elife*, 5:e11752, 2016.

26 David Weese, Manuel Holtgrewe, and Knut Reinert. RazerS 3: faster, fully sensitive read mapping. *Bioinformatics*, 28(20):2592–2599, 2012.

27 Thomas D Wu and Serban Nacu. Fast and SNP-tolerant detection of complex variants and splicing in short reads. *Bioinformatics*, 26(7):873–881, 2010.

28 Andrew Yates, Wasiu Akanni, M Ridwan Amode, Daniel Barrell, Konstantinos Billis, Denise Carvalho-Silva, Carla Cummins, Peter Clapham, Stephen Fitzgerald, Laurent Gil, et al. Ensembl 2016. *Nucleic acids research*, page gkv1157, 2015.

29 Matei Zaharia, William J Bolosky, Kristal Curtis, Armando Fox, David Patterson, Scott Shenker, Ion Stoica, Richard M Karp, and Taylor Sittler. Faster and more accurate sequence alignment with SNAP. *arXiv preprint arXiv:1111.5572*, 2011.

30 Mohsen Zakeri, Avi Srivastava, Fatemeh Almodaresi, and Rob Patro. Improved data-driven likelihood factorizations for transcript abundance estimation. *Bioinformatics*. (Proceedings of ISMB 2017, in press).