BIOINFORMATICS

Vol. 24 ECCB 2008, pages i174–i180
doi:10.1093/bioinformatics/btn300

Optimal spliced alignments of short sequence reads

Fabio De Bona[1], Stephan Ossowski[2], Korbinian Schneeberger[2] and Gunnar Rätsch[1,*]
[1]Friedrich Miescher Laboratory, Max Planck Society, Spemannstr. 39, 72076 Tübingen and [2]Max Planck Institute for Developmental Biology, Spemannstr. 35, 72076 Tübingen, Germany

ABSTRACT

Motivation: Next generation sequencing technologies open exciting new possibilities for genome and transcriptome sequencing. While reads produced by these technologies are relatively short and error prone compared to the Sanger method their throughput is several magnitudes higher. To utilize such reads for transcriptome sequencing and gene structure identification, one needs to be able to accurately align the sequence reads over intron boundaries. This represents a significant challenge given their short length and inherent high error rate.

Results: We present a novel approach, called *QPALMA*, for computing accurate spliced alignments which takes advantage of the read's quality information as well as computational splice site predictions. Our method uses a training set of spliced reads with quality information and known alignments. It uses a large margin approach similar to support vector machines to estimate its parameters to maximize alignment accuracy. In computational experiments, we illustrate that the quality information as well as the splice site predictions help to improve the alignment quality. Finally, to facilitate mapping of massive amounts of sequencing data typically generated by the new technologies, we have combined our method with a fast mapping pipeline based on enhanced suffix arrays. Our algorithms were optimized and tested using reads produced with the *Illumina Genome Analyzer* for the model plant *Arabidopsis thaliana*.

Availability: Datasets for training and evaluation, additional results and a stand-alone alignment tool implemented in C++ and python are available at http://www.fml.mpg.de/raetsch/projects/qpalma.

Contact: Gunnar.Raetsch@tuebingen.mpg.de

1 INTRODUCTION

Next generation (NG) sequencing technologies are able to generate huge amounts of DNA sequence reads at a fraction of the cost of Sanger sequencing. While the human genome project cost several hundred million US dollars, new sequencing technologies like Roche/454's FLX Genome Sequencer are able to sequence a human genome with no more than 1 million US dollars. Recently introduced sequencing technologies like Illumina's Solexa sequencing technology or ABI's SOLiD are able to generate the same amount of sequences with an order of magnitude lower costs. However, these technologies also come with certain limitations in particular concerning the read length and the rate of sequencing errors. These characteristics make their use for genome and transcriptome sequencing considerably more challenging. So far, most efforts were spent in developing methods for analysing sequence reads from *genomic DNA*, for instance efficient alignments of reads to reference genomes for genome resequencing and also *de novo* genome assembly (e.g. Hillier *et al.*, 2008; Sundquist *et al.*, 2007; Wold and Myers, 2008; Zerbino and Birney, 2008). While the latter techniques seem in principle be useful for *transcriptome analysis*, they typically do not use the genomic sequence for guiding the assembly and are additionally faced with alternative transcripts which can result in assembly errors.

For EST and cDNA sequences one therefore resorts to a different strategy: instead of assembling the sequences before aligning them to the genome, one first aligns the single reads to the genome and then merges the alignments to infer gene structures. Many methods have been developed to solve the so-called spliced alignment problem of aligning spliced RNA sequences to the genome (Florea *et al.*, 1998; Gelfand *et al.*, 1996; Kent, 2002; Schulze *et al.*, 2007; Slater and Birney, 2005; Usuka *et al.*, 2000; Zhang and Gish, 2006). These methods are efficient and accurate if the sequence blocks (exons) are sufficiently long and are highly similar to the genomic sequence. Reads from NG sequencing techniques typically do not have either of the two properties. For instance, if a single read of length 30 nt spans over two exons, it is not very unlikely that the shorter part covering one exon is not longer than just 5 nt (>16%). Moreover, assuming a substitution error rate of 5% it is very likely that there is at least one mismatch within these 5 nt. This represents a substantial challenge to alignment algorithms such as *blat* for correctly aligning such reads.

In this work we aim to develop a method exploiting all available information to accurately align as many as possible spliced sequence reads to the genome. As information sources there is not only the DNA sequence of the read and the genome, but also quality information associated with the read and predictions about potential splice sites within the genome. In our previous work we already proposed a method taking advantage of splice site predictions (Schulze *et al.*, 2007). In this work we extend this method to also benefit from the read's quality scores. This information can help to decide at which positions one can expect to observe mismatches and subsequently contribute to the identification of the correct alignment.

Rather accurate methods for relating the quality score to the error probability which can then be used in some probabilistic model have been proposed for Sanger sequencing (Li *et al.*, 2004; Mott, 1998). However, such measures are less well developed and accurate for NG sequencing techniques. We therefore propose an alternative method for taking quality scores of a sequence into account. The idea is to learn, in a supervised manner, how to score quality information, splice site predictions and sequence identity based on a representative set of sequence reads with *known* alignments. The algorithm is based on extensions of the well-known Smith-Waterman algorithm using more sophisticated parametrized scoring functions. The idea is to tune the parameters of the scoring functions such that the true alignment does not only achieve a large score (to be 'most likely'), but also that all other alignments score considerably

---

[*] To whom correspondence should be addressed.

© The Author 2008. Published by Oxford University Press. All rights reserved. For Permissions, please email: journals.permissions@oxfordjournals.org

lower than the true alignment (to obtain a 'large margin between the alignments'). Similar ideas are used in other large margin algorithms such as Support Vector Machines (SVMs) (Müller *et al.*, 2001; Schölkopf and Smola, 2002; Vapnik, 1995) and Boosting (Freund and Schapire, 1997; Meir and Rätsch, 2003). The resulting scoring function can then be used to obtain the best scoring alignment via the extended Smith–Waterman algorithm.

To train and evaluate our method we need a set of representative sequences for which we know the true alignment. The most representative set of sequences would be obtained by sequencing short reads from the transcriptome. However, in this case we would not necessarily know the correct alignment: Only in the simplest cases we could come up with an accurate alignment based on standard alignment techniques. This set of sequences would not be suitable for evaluating a method aimed to be more accurate than a standard alignment technique, since one would not be sure which method made a mistake. We therefore chose to use short *genomic* sequence reads produced with the *Illumina Genome Analyzer* for generating *in silico* spliced sequence reads based on the genome annotation. The idea is to consider all genomic sequence reads overlapping with two consecutive exon boundaries (according to the annotation). If a read covers the end of the first exon, it can be combined with a read covering the start of the other exon. Using this merge operation, we can generate reads that basically look like transcriptome reads produced on the same platform particularly with regard to read error probability and per base quality, for which we exactly know the correct spliced alignment.

In a typical application scenario one needs to align millions of short sequence reads against the genome. In this case the direct application of the extended Smith–Waterman algorithm for alignment against the whole genome is not feasible. We therefore propose to combine our method with a fast suffix array-based approach to identify a seed for the alignment. A read will in the great majority of the cases span over one or two exons. Hence, the longer part of the read is going to be long enough to be found quite unambiguously in the genome if allowing for a small number of substitutions or indels. For each such seed we can use *QPALMA* to align the read to the genomic regions surrounding the seed position to identify the other part of the match. In some cases there will be several seeds. Then *QPALMA*'s scoring function can be used to decide which seed leads to the correct alignment. This combined strategy will allow us to efficiently align even very large numbers of reads identifying their spliced alignments.

The article is structured as follows: In Section 2 we first describe the different parts of *QPALMA*, describe the generation of *in silico* spliced reads derived from a *Illumina Genome Analyzer* and finally describe the pipeline for efficient alignment of large sequence sets. In Section 3 we will show the power of our approach while illustrating that each information source leads to additional increases in performance. Finally we show first results using the proposed alignment pipeline and conclude this work with a discussion in Section 4.

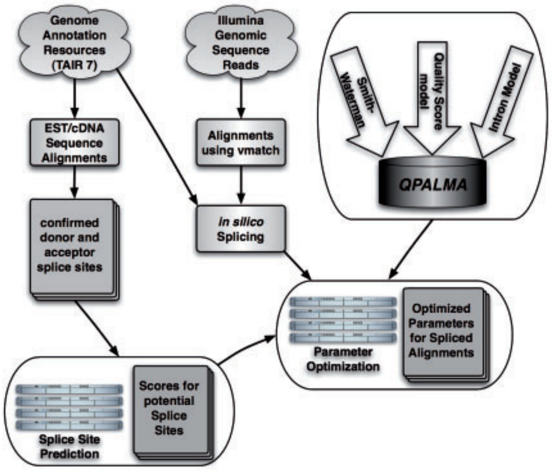
Genome Annotation Resources (TAIR 7)
Illumina Genomic Sequence Reads
EST/cDNA Sequence Alignments
Alignments using vmatch
confirmed donor and acceptor splice sites
in silico Splicing
Smith-Waterman
Quality Score model
Intron Model
QPALMA
Parameter Optimization
Optimized Parameters for Spliced Alignments
Splice Site Prediction
Scores for potential Splice Sites

Fig. 1. Work-flow for training *QPALMA*: Confirmed donor and acceptor sites are used to train a SVM-based splice site predictor (Sonnenburg *et al.*, 2007). Short sequence reads (36 nt) obtained from an Illumina 1G sequencing machine and the *A.thaliana* genome annotation (TAIR 7) were used to derive a training and evaluation set. *QPALMA* generalizes the Smith–Waterman algorithm by including sequence quality information and an intron model considering splice site predictions as well as intron length information to learn how to produce optimally spliced alignments.

2 METHODS

In the following sections we will describe our method, called *QPALMA*, consisting of three independent parts: the splice site prediction model, the dynamic programming algorithm and the optimization of the scoring function thereby solving the so-called *inverse alignment problem* (e.g. Kececioglu and Kim, 2006). Additionally, we will outline the *in silico* generation of spliced reads for training based from genomic reads and propose a pipeline combining enhanced suffix arrays, based on *vmatch* (Abouelhoda *et al.*, 2002) and *QPALMA* to align millions of short transcriptome reads as for instance generated by NG sequencing techniques.

2.1 Splice site predictions

For predicting splice sites, one first needs a dataset of known acceptor and donor splice sites as well as suitable decoy sequences. Such sequences can be obtained by aligning EST and cDNA sequences using a standard alignment program suitable for long sequence reads and spliced alignments [e.g. *blat* (Kent, 2002) or PALMA (Schulze *et al.*, 2007)]. High quality alignments can be used to confirm acceptor as well as donor splice sites and aligned genomic regions can be used for sampling decoy site by considering all sites with consensus `AG` and `GT/C`. For each site we consider a sequence window of 141 nt around the splice site, which is used for classification into splice site versus decoy (separately for acceptor and donor sites). To learn a classifier one may use SVMs with the so-called 'weighted degree' kernel (Rätsch *et al.*, 2006; Sonnenburg *et al.*, 2002). This kernel computes the similarity between two sequences $\mathbf{s}$ and $\mathbf{s}'$ by considering substrings occurring in both strings up to length d. In Sonnenburg *et al.* (2007) we have done these steps and also provided genome-wide predictions for several organisms at `ftp://ftp.tuebingen.mpg.de/fml/behr/splicing`. In this work we use these predictions for *Arabidopsis thaliana*.

2.2 Extensions of the Smith–Waterman algorithm

In this section we will discuss three extensions of the well-known Smith–Waterman algorithm (Smith and Waterman, 1981) for local sequence alignments which can be used in combination with the parameter estimation algorithm outlined in the next section.

2.2.1 The classical algorithm The classical deterministic and exact alignment algorithm is the Smith–Waterman algorithm and is based on dynamic programming. Its running time is $\mathcal{O}(m \cdot n)$, where m is the length of the short read S_E, and n is the length of the DNA sequence S_D. It builds up a $m \cdot n$ matrix and hence has the same space complexity.

The main idea of the algorithm is to compute a local alignment by determining the maximum over all alignments of all prefixes $S_E(1:i) := (S_E(1),\dots,S_E(i))$ and $S_D(1:j) := (S_D(1),\dots,S_D(j))$ of the two sequences S_E and S_D, while allowing for unaligned starts of the sequences (details below). An alignment is given by a sequence of pairs (a_r, b_r), $r = 1,\dots,R$, where $R \le m+n$ depends on the alignment and $a_r, b_r \in \Sigma := \{A, C, G, T, N, -\}$. A pair consists either of the two corresponding letters of the two sequences or a single letter in one sequence paired with a gap in the other sequence. The alignment is scored using a substitution matrix M, which we interpret as a function $M : \Sigma \times \Sigma \to \mathbb{R}$. Then the score for the alignment $\mathcal{A} = \{(a_r, b_r)\}_r$ is simply $\sum_r M(a_r, b_r)$.

We define $V(i_2, j_2)$ to be the score of the best alignment of prefixes $S_E(i_1 : i_2)$ and $S_D(j_1 : j_2)$ for the best choice of starting positions i_1 and j_1 on sequences E and D, respectively. The best local alignment can be obtained by finding the maximal entry in the matrix V determining i_2 and j_2 as the ends of the alignment. The matrix V can be computed using the following recurrence formula (for all $i = 1,\dots,m$ and $j = 1,\dots,n$):

$$V(i,j) = \max \begin{cases} 0 \\ V(i-1,j-1) + M(S_E(i), S_D(j)) \\ V(i-1,j) + M(S_E(i), '-') \\ V(i,j-1) + M('-', S_D(j)) \end{cases} \tag{1}$$

The recurrence is initialized with $V(0,0) := 0$, $V(i,0) := 0$ and $V(0,j) := 0$ for all $i = 1\dots m$ and $j = 1\dots n$. There are four possibilities: (a) $S_E(1:i)$ and $S_D(1:j)$ are unaligned; (b) $S_E(i)$ and $S_D(j)$ are aligned to each other (possibly a mismatch); (c) $S_E(i)$ is aligned to a gap in the DNA sequence and (d) $S_D(j)$ is aligned to a gap in the short sequence. In the original algorithm there are only these four possibilities and one can straightforwardly fill the matrix from left to right and top to bottom to finally compute the maximum over all elements in V. The optimal alignment can then be obtained by backtracking (Durbin *et al.*, 1998).

2.2.2 *Extension 1: Quality Scores*

In a first step we extend the scoring system algorithm to take quality information of the short sequence read into account [a similar idea was proposed in a simpler form in (Mott, 1998)]. The idea is relatively straightforward: So far $M : \Sigma \times \Sigma \to \mathbb{R}$ only scored an alignment based on matches, mismatches or indels. Here, we define M to be a function of the two aligned letters as well as the *quality score* of the read at the corresponding position, i.e. $M : \Sigma \times \mathbb{R} \times \Sigma \to \mathbb{R}$.[1] Note that we only have quality information available for positions not corresponding to a gap on the short read. Hence, the functions $M('-', \cdot, b)$ $(b \in \Sigma)$ can be considered as a constant. Given this scoring function we can now extend the recurrence formula (for all $i = 1,\dots,m$ and $j = 1,\dots,n$):

$$V(i,j) = \max \begin{cases} 0 \\ V(i-1,j-1) + M(S_E(i), Q_E(i), S_D(j)) \\ V(i-1,j) + M(S_E(i), Q_E(i), '-') \\ V(i,j-1) + M('-', \cdot, S_D(j)) \end{cases}, \tag{2}$$

where $Q_E(i)$ is the quality score of the short read at position i.

Please note that this algorithm has the same computational complexity as the original Smith–Waterman algorithm ($\mathcal{O}(mn)$). However, it uses a more complex scoring that may depend on the sequencing technology used. We chose to represent the scoring function as a set of one dimensional functions—one for every match-pair. Hence, we require 6*x*6 such functions ($M_{a,b}(q) := M(a,q,b)$, $a, b \in \Sigma$), out of which 6 are constant (corresponding to a gap on the short read).

2.2.3 *Extension 2: Splice Sites*

The Smith–Waterman algorithm aligns two sequences based on single base pairs and does not distinguish between exons and introns. In Schulze *et al.* (2007) we therefore proposed to extend the Smith–Waterman algorithm to better model introns. The previously proposed algorithm required considerably more computing time: $\mathcal{O}(mnL)$ operations, where L is the maximal length of the intron. Given the large number of reads to be aligned, it is highly desirable to keep the required computing time low. Hence, we propose an alternative formulation that does model *splice sites* but only scores the intron length by an *affine function* while only requiring $\mathcal{O}(mn)$ operations. The idea is to maintain an additional recurrence matrix W used to keep track of the intron boundaries. We use the following recurrence formulas:

$$V(i,j) = \max \begin{cases} 0 \\ V(i-1,j-1) + M(S_E(i), Q_E(i), S_D(j)) \\ V(i-1,j) + M(S_E(i), Q_E(i), '-') \\ V(i,j-1) + M('-', \cdot, S_D(j)) \\ W(i,j-1) + \hat{f}_{acc}(j-1) \end{cases} \tag{3}$$

and

$$W(i,j) = \max \begin{cases} V(i,j) + g_o + \hat{f}_{don}(i+1) \\ W(i,j-1) + g_e \end{cases}, \tag{4}$$

where g_o and g_e are the intron opening and extension scores. The $g_{don}(i)$ and $g_{acc}(i)$ are scoring functions for splice sites at position i in the sequence, which take the form $\hat{f}_{acc}(i) := f_{acc}(g_{acc}(i))$ and $\hat{f}_{don}(i) := f_{don}(g_{don}(i))$. Here $g_{acc}(i)$ and $g_{don}(i)$ are the splice site score precomputed by the splice site SVMs (cf. Section 2.1) and $f_{acc}, f_{don} : \mathbb{R} \to \mathbb{R}$ are scoring function appropriately transforming the SVM outputs.[2]

It can easily be verified that for each identified intron between positions k and j, the above recurrences add a score depending on its length $j-k$ and the splice site strengths given as follows:

$$f_I(k,j) = g_o + g_e(j-k-1) + f_{don}(g_k) + f_{acc}(g_j).$$

2.2.4 *Extension 3: Non-affine Intron Length Model*

If we would like to score the intron length with an arbitrary function $f_L : \mathbb{N}_+ \to \mathbb{R}$, i.e.

$$f_I(k,j) = f_L(j-k) + f_{don}(g_k) + f_{acc}(g_j),$$

the recurrence can be implemented as follows:

$$V(i,j) = \max \begin{cases} 0 \\ V(i-1,j-1) + M(S_E(i), Q_E(i), S_D(j)) \\ V(i-1,j) + M(S_E(i), Q_E(i), '-') \\ V(i,j-1) + M('-', \cdot, S_D(j)) \\ \max\limits_{j-L_{\max} \le k \le j-1} (V(i,k) + f_I(k,j)) \end{cases}, \tag{5}$$

where $L_{\max}$ is the maximal intron length. This recurrence has been proposed in Schulze *et al.* (2007) in a similar form and is considerably more computationally expensive than the previous one: every step involves finding the optimal intron start ($\mathcal{O}(L_{\max})$), leading to the complexity of the dynamic programming algorithm of $\mathcal{O}(mnL_{\max})$. For long introns this approach seem computationally infeasible.

For completeness we need to extend our notation for alignments with introns. We use again alignment pairs $\mathcal{A} = \{(a_r, b_r)\}_r$, but extend the alphabet for a_r to $\Sigma \cup \{+\}$ ('intron sequence missing') and for b_r to $\Sigma \cup \{*\}$ ('intron sequence'). Note that b_r should only contain strings of length greater than one if $a_r = '+'$. Then the score $f(\mathcal{A})$ for an alignment $\mathcal{A}$ with intron is computed as before, i.e. $\sum_r M(a_r, b_r)$, except when $a_r = +$: In this case the intron score function $f_I(\cdot,\cdot)$ is used to score the corresponding intron.

---

[1] It is straightforward to extend this to the case where quality information is available for both sequences.

[2] When there is no donor consensus at position i, then we define $f_{don}(g_{don}(i)) := -\infty$ (analogously for $f_{acc}(g_{acc}(i))$).

2.3 Solving the inverse alignment problem

In the previous section we assumed that the functions f_{acc}, f_{don} and f_L as well as the 36 scoring functions for the quality as represented by M were given. We now describe an algorithm to determine these parameters based on the training set of sequences (with quality scores) and their true alignments. This algorithm has been proposed before in Schulze *et al.* (2007) based on the original ideas in Altun *et al.* (2003) for slightly the simpler case without quality information. We therefore only outline the basic idea of the algorithm—more details are found in Schulze *et al.* (2007).

2.3.1 Parametrization Each of the functions to be determined is one dimensional. It therefore suffices to use a simple piecewise linear model: let s be the number of supporting points, x_i (satisfying $x_i < x_{i+1}$) and y_i their values, then the piecewise linear function is defined by

$$f(x) = \begin{cases} y_1 & x \le x_1 \\ \frac{y_i(x_{i+1}-x)+y_{i+1}(x-x_i)}{x_{i+1}-x_i} & x_i \le x \le x_{i+1} \\ y_s & x \ge x_s \end{cases} . \qquad (6)$$

After having appropriately chosen supporting points on the x-axis we only need to optimize the corresponding y-values.

Note that given the support points and their corresponding y-values, the alignment function $f(\mathcal{A})$ for an alignment $\mathcal{A}$ is fully specified. Moreover, by design the scoring function is linear in all parameters, i.e. the y-values. Hence, it can be written as $f(\mathcal{A}) = \Phi(\mathcal{A})^\top \theta$ for an appropriately defined Φ, where θ is the vector of parameters corresponding to the y-values at the support points of all functions.

2.3.2 Optimization For training we are given a set of N true alignments $\mathcal{A}_i^+, i = 1, \ldots, N$. The goal is to find the parameters $\boldsymbol{\theta}$ of the alignment scoring function f such that the difference of scores $f_\theta(\mathcal{A}_i^+) - f_\theta(\mathcal{A}^-)$ is large for *all* wrong alignments $\mathcal{A}^- \ne \mathcal{A}_i^+$. This can be done by solving the following convex optimization problem:

$$\min_{\boldsymbol{\xi} \ge \mathbf{0}, \boldsymbol{\theta}} \quad \frac{1}{N} \sum_{i=1}^{N} \xi_i + C\mathbf{P}(\boldsymbol{\theta}) \qquad (7)$$

$$\text{s.t.} \quad f_\theta(\mathcal{A}_i^+) - f_\theta(\mathcal{A}^-) \ge 1 - \xi_i \quad \forall i \text{ and } \mathcal{A}^- \ne \mathcal{A}_i^+ .$$

Here we introduced so-called slack-variables ξ_i to implement a soft-margin (Cortes and Vapnik, 1995), i.e. to allow for a few misaligned examples. Additionally, we use a regularization term $\mathbf{P}(\boldsymbol{\theta})$ to avoid over-fitting by preferring smooth piece-wise linear functions [see Schulze *et al.* (2007) for details]. The parameter C controls the trade-off between smoothness and fit to the training data. Note that the above optimization problem has too many constraints to be solved directly. In Schulze *et al.* (2007) we give a detailed description of an algorithm based on *column generation* for iteratively solving such optimization problems. This algorithm requires computing best-scoring alignments for suboptimal parameter settings. This can be done by using the corresponding version of the Smith–Waterman algorithm.

2.4 *In silico* generation of spliced reads

We sequenced the *A. thaliana* reference genome using the *Illumina Genome Analyzer* producing reads of length 36.[3] From the 80 344 405 reads that passed initial quality filtering, 71 580 097 reads (89%) could be aligned against the *A. thaliana* reference sequence using *vmatch* [Abouelhoda *et al.* (2002), available at `http://www.vmatch.de`] resulting in an average coverage of 16. For each read only the best matches (measured by E-value) were considered allowing up to four mismatches or three gaps. More than 67% of the reads align without error, 92% with less than three mismatches.

Based on the TAIR 7 genome annotation (available at http://www.arabidopsis.org), we identified pairs of reads that can be combined to a 36 nt sequence read spanning over an annotated intron.[4] We additionally require that the quality scores around the junction are similar—otherwise the method might identify the junction by judging the differences in quality scores which we would like to avoid. All possible pairs around a junction were considered, but each read was used at most once in a merged pair. We also generated a list of reads completely contained within annotated exons as unspliced reads. This lead to 246 586 merged ('spliced') and 2 339 584 original ('unspliced') reads aligned to the forward strands of chromosome 1.

2.5 An alignment pipeline against whole genomes

Computation of optimal alignments is quite time consuming given the large number of reads produced by NG sequencing approaches. We therefore designed a multi-step approach combining a fast matching algorithm based on enhanced suffix arrays (*vmatch*, Abouelhoda *et al.*, 2002) for initial read mapping and the proposed optimal alignment algorithm based on dynamic programming (*QPALMA*) for high quality detection of splice sites. An overview of this pipeline is given in Figure 2.

In a first step we use *vmatch* to find global alignments of all reads (with at most two mismatches) against the genome to identify the large fraction of unspliced reads. The set of successfully aligned reads presumably still contains a small fraction of spliced reads where the intron is near the read boundary. To identify such reads we used *QPALMA*'s scoring function to develop a filter that can quickly decide whether the read is spliced or not.[5] The idea is to compute the scoring function of the alignment returned by *vmatch* and compare it with the scores of possible spliced alignments:

- all combinations of putative donor splice sites within the read and acceptor splice sites ≤2000 nt downstream of the read, and
- all combinations of putative acceptor splice sites within the read and donor splice sites ≤2000 nt upstream of the read.

If there exists a combination with larger score than the unspliced alignment, then the filter predicts the read to be *spliced* and *unspliced* otherwise.

Reads that cannot be aligned in the first *vmatch* round fall into two categories: low quality reads or spliced reads. In the second step these left-over reads as well as reads that were predicted to be spliced by the *QPALMA* filter are aligned with about half the read's length using *vmatch* to produce seeds for further alignment steps. We use a reasonably sized window (2000 nt) around each 'hit' to align using *QPALMA*. If a seed exists at several genomic locations, then it is crucial that we report the correct alignment position. This is done by comparing *QPALMA*'s alignments scores for each seed and selecting the seed with the highest score.

The error rate of the above pipeline is determined by three factors: (1) Can the *QPALMA* filter correctly decide whether a read is unspliced or spliced, even if there exists reasonable full-length unspliced alignment? (2) Can *QPALMA* correctly identify the correct *vmatch* seed? (3) Can *QPALMA* identify the correct exon boundaries?

As shown in Figure 3, the error rates will be highest for reads were the intron is very near the read boundary. These errors are due to the resulting very short exons leading to ambiguities that result in either falsely unspliced alignments [important for (1)] or spliced alignments with a wrong placement of the shorter exon [important for (3)]. While the first case is rather difficult to identify, the second one is easily identifiable by the predicted intron position. Hence, a possible strategy to reduce the number of alignment errors would be to ignore all spliced alignments that are too close to the read boundaries.

Alternatively, we propose to align all reads against the flanking sequences of each predicted intron (unspliced global alignment). Ideally we will find several other alignments of reads confirming the putative intron. The total number of reads confirming the intron without mismatches near the intron

---

[3] Data were provided by Detlef Weigel, Richard Clark and Christa Lanz, personal communication.

[4] If a gene has several annotated transcripts, we only considered the first one.

[5] In the filter, we only consider a small set of possible alignments. This can be computed many times faster than the dynamic program.

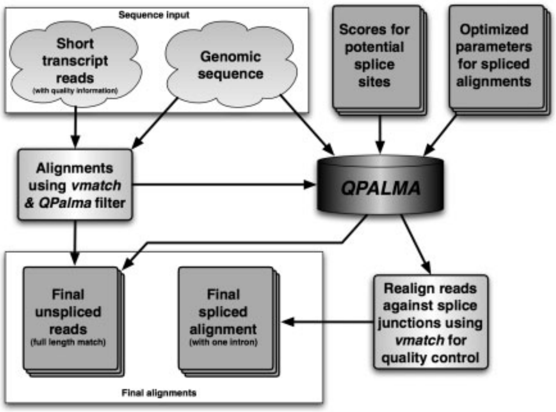
Sequence input
Short transcript reads
(with quality information)
Genomic sequence
Scores for potential splice sites
Optimized parameters for spliced alignments
Alignments using *vmatch* & *QPalma* filter
QPALMA
Final unspliced reads
(full length match)
Final spliced alignment
(with one intron)
Realign reads against splice junctions using *vmatch* for quality control
Final alignments

Fig. 2. Proposed alignment pipeline for using *QPALMA*: Short sequence reads are first aligned with *vmatch* to identify unspliced reads. Unmapped or potentially spliced reads are aligned again to identify reads of at least half of the read length to find seeds for use with *QPALMA*. For each seed position *QPALMA* aligns the read and returns a score. The best scoring alignment is returned as the spliced alignment of the read, if the intron can be confirmed at least two times by another read.

boundary can be used as a confidence measure for the intron, which we call the *remapping score*. We expect that the error drops drastically if one requires high remapping scores for introns. Hence, if the transcriptome coverage is high enough, one can significantly improve upon the single read alignment accuracy.

3 EXPERIMENTS

3.1 Comparison of Smith–Waterman extensions

In this section we compare the different extensions of the Smith–Waterman algorithm (Section 2.2) in combination with the learning algorithm outlined in Section 2.3. We trained the algorithm using 10 000 sequences with known alignments (as described in Section 2.4) in eight different combinations resulting from switching on and off the three different scorings: quality information, splice site predictions[6] and intron length.

We test our algorithm on 30 000 sequences different from the training set for an unbiased estimation of *QPALMA*'s accuracy on unspliced reads. We compute the fraction of reads that have been accurately aligned at all four boundaries (start and end of first and second exon). The results are given in Table 1. We can observe that the algorithm without using any additional information has the largest error rate of about 14.19%. If we include quality score information, then the error rate reduces by 0.7% to 13.49%—a moderate improvement showing that the quality information can indeed help to identify the correct alignment. If we additionally include the splice site predictions, then the error rate drastically reduces by 10.68% to 2.81%. This shows that it is not sufficient to just consider the splice site consensus to achieve a good alignment: an accurate splice site detector can significantly decrease the error rate. Finally, we also include the intron length into the model. It leads to an additional improvement of about 1.03% to 1.78%. We can therefore conclude that all three components—quality scores, splice site predictions and intron length—help to reduce the alignment error rate.

Table 1. The alignment error rates for different versions of *QPALMA* (without the *vmatch* seeds) trained on 10 000 *in silico* spliced reads: with and without read quality information, intron length scoring and splice site predictions, respectively

Quality information	Intron length	Splice site pred.	Error rate (%)
−	−	−	14.19
+	−	−	13.49
−	+	−	9.96
+	+	−	9.68
−	−	+	3.16
+	−	+	2.81
−	+	+	1.94
+	+	+	**1.78**

Evaluation was done on 30 000 reads aligned to their genomic origin including 1500 nt up- and downstream sequence. A read is counted as correctly aligned if the intron boundaries as well as the alignment ends exactly matched the template. The lowest error is obtained when using all three sources of information. The error is significantly lower than for all other models (significance level 95%).

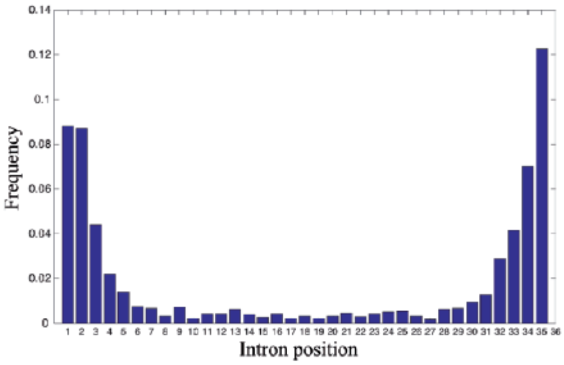

Fig. 3. Error rate versus intron position: on the x-axis is the intron position (equal to the length of the first exon). On the y-axis is the alignment error rate for reads with introns at the given position (estimated from 120 000 spliced reads). We observe that reads leading to very short exons have the largest error rate and contribute most to the overall error.

In the generation of the spliced sequences we also included many cases where the smaller part (in one of the exons) is very short. In Figure 3 we show the error rate separately for every intron position: If the intron is near the boundary, then one exon is very short. Not entirely surprising, we observe that most mistakes are made for cases where the smaller part is 4 nt or smaller, especially since it is known that there is a drastic increase of errors at the end of reads. If we exclude such cases from the evaluation, then the total error rate on the test set reduces from 1.78% to about 0.51%. It therefore seems to be a reasonable strategy not to consider alignments where one of the exon parts is not longer than 4 nt (in our case about 20% of all spliced reads).

3.2 Illustration of the learning result

In Figures 4 and 5, we have displayed the parameters that are the result of our learning algorithm for the case with quality information, splice site predictions and intron length. Shown are the piece-wise

---

[6] When we do not use splice site predictions, we still require the presence of the `GT/AG` consensus at the intron boundaries.

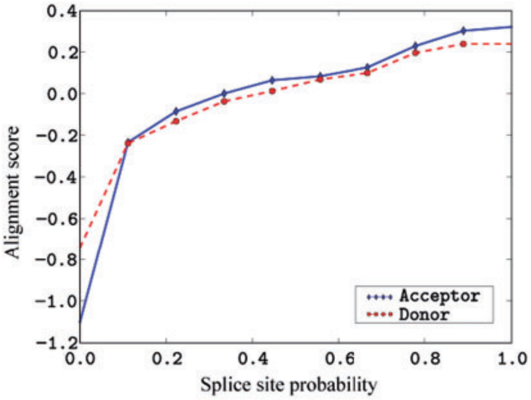

Fig. 4. Learned piece-wise linear functions to score acceptor and donor splice sites: High probabilities for splice sites contribute strongly to the alignment score.

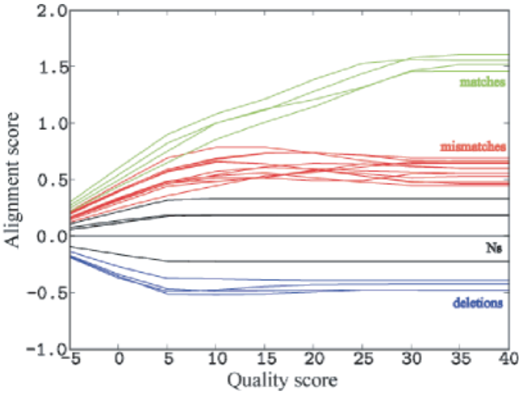

Fig. 5. Learned piece-wise linear functions to score matches (4 × green), mismatches (12 × red), `N`'s (5 × black) and deletions (5 × blue). Larger quality scores for matches contribute stronger to the alignment score. Mismatches with larger quality score receive a lower alignment score than with medium sized quality scores.

linear functions scoring splice sites (Fig. 4) as well as matches, mismatches, `N`'s (on DNA) and deletions (on DNA). We observe that the quality scoring functions cluster into the four groups. Within the groups the variation is relatively small indicating that there are no strong biases in the quality scores for certain error combinations in our data.

3.3 Results on whole genome alignments

We finally present results on a relatively large dataset using the pipeline described in Figure 2. The dataset used is a subset of the 71×10^6 reads mentioned before and contains a total number of 2.98×10^6 reads from the transcribed part of the positive strand of chromosomes I–V (as annotated), out of which about 10% are spliced (285 530 reads). Please note that we omitted the 10 000 spliced reads used for training *QPALMA* in following analysis.

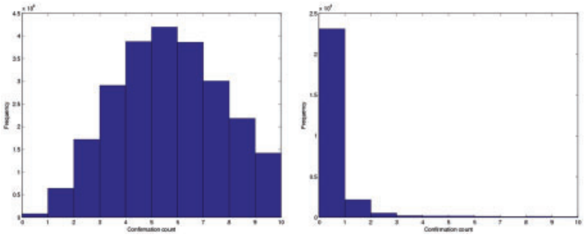

Fig. 6. Remapping score distribution for correctly and incorrectly aligned reads. Most correctly aligned reads can be confirmed by at least one other read, while incorrectly aligned reads are rarely confirmed more than once.

3.3.1 Data flow In a first step we use *vmatch* to align all reads (cf. Fig. 2). The alignment step was parametrized to consider the whole 36-mer and to allow for two mismatches and no gaps. From this *vmatch* run we obtained a set of reads that could be aligned of 2 511 338 and 471 009 reads that could not be aligned with the given mismatches and read length. Ideally the first set of aligned reads would only contain unspliced reads. The previously described *QPALMA* filter was able to identify 43 148 reads in this set that are actually spliced (8964 false positives out of 2.51×10^6). The 471 009 unalignable reads in the first *vmatch* round are aligned again using *vmatch* with at least 18 nt alignment length and at most one mismatch. In this run, for 12 864 (3428 spliced and 9436 unspliced) reads no alignment could be found on the positive strand of chromosomes I–V.[7] In the next step we used *QPALMA* to align the 52 112 reads found by the *QPALMA* filter together with the alignable part (458 145) in the second *vmatch* round: 230 795 reads were determined to be unspliced and 269 631 reads to be spliced.

3.3.2 Pipeline accuracy Out of the 2 696 817 unspliced reads in total, 14 566 reads were either without seed on the positive strand of chromosomes I–V (9566) or aligned to the wrong genomic position (5000). A total of 18 696 were aligned spliced, leading to a total error rate of unspliced reads of 1.2%. From the 275 530 spliced reads, 2556 and 3675 were aligned unspliced by *QPALMA* filter and *QPALMA*, respectively, and 8189 were aligned wrongly, i.e. with wrong exon boundaries. This leads to an error rate of spliced reads of 5.2%. Out of the latter 8189 sequences a large fraction (7129) were aligned wrongly due to a wrong *vmatch* seed position. If we exclude such reads, we observe that 7291 spliced reads are misaligned due to *QPALMA*, leading to an error rate of 2.6%. This is only slightly higher than previously observed (cf. Table 1). In total, 51 138 reads out of the 2 982 347 reads where either unaligned or aligned incorrectly, leading to an error rate of the pipeline of 1.7%.

If we only consider spliced alignments that can be confirmed by at least two other reads,[8] the number of wrongly aligned spliced reads drops from 8189 to 1990, while one loses about 10% of the spliced reads (cf. Fig. 6). Out of the 18696 unspliced reads predicted as spliced, not a single one remained. Hence, the total error rate reduces to 0.88%.

It can be observed that most errors are induced by wrong or missing *vmatch* seed positions (0.7%). Also, often (0.6%) unspliced reads were predicted as spliced. The latter errors can be reliably detected by requiring three reads confirming a spliced alignment.

---

[7] Note that, we allowed up to four mismatches for the initial *vmatch* alignment to derive the *in silico* dataset. In the pipeline, we only allowed two mismatches, which leads to a small fraction of reads with no match.

[8] For each splice site 34 nt 5′ to the donor side and 34 nt 3′ to the acceptor site have been concatenated and were realigned using all reads with a maximum of two mismatches (no indels). A splice site has been rejected if not more than three reads covered the 4 nt around the splice site without mismatches.

Table 2. CPU times of the different processing steps on a single CPU core

Step	Total time	Number of reads	Reads/sec
Vmatch runs	≈4 h	2 586 170	179
QPALMA filter	≈17 min	2 180 858	417
QPALMA prediction	≈8 h	441 579	15
Remapping score	≈10 min	249 001	733
QPALMA training	≈6 h	10 000	0.5

Training is only required once per genome.

3.3.3 Computing time The total time needed for the running the pipeline on ≈2.5 million reads is given in Table 2 (on a single CPU core). If scaled to the about 71 million reads from the whole genome, the alignment using this pipeline would take about 400 h of computing time. Distributed on 20 CPU nodes it is just about one day of computing time.

The total computing time is dominated by the *QPALMA* prediction step (≈8 h). This can be speedup by either improving the implementation (e.g. by exploiting special CPU features, as in *SHRiMP*) or by using the approximation of the *QPALMA* filter for computing the alignment (≈30 times faster). Training of *QPALMA* took about 6 h on 10 000 reads. However, this is only required once per genome and sequencing platform and does not matter for the application of the pipeline.

4 DISCUSSION

We have presented a novel approach to solve the difficult task of aligning short sequence reads as generated by NG sequencing techniques over exon boundaries. We were able to successfully exploit all available information sources—the read including its quality information, splice site predictions, the intron length and, of course, the genome—each significantly contributing to decreasing the alignment error rate. If we only consider spliced reads that significantly overlap into the exon (> 4 nt), the error rate is as small as 0.5%. For reads that only overlap 1–2 nt into the next or previous exon, the error rate can be as high as 12%, which is as expected, as random matches are quite likely. If the transcriptome coverage is high enough, the proposed remapping score can be used to find the doubtful alignments in order to significantly reduce the alignment error rate.

So far we have only used *QPALMA* for the *Illumina* sequencing platform. The same approach is expected to work reasonably well also for other NG sequencing platforms, as all parameters are tuned during training to work well for the considered platform. The only precondition is that there are genomic reads available that can be used to generate artificially spliced reads for training. *QPALMA* also learns how to score the quality information of the read. It would be straightforward to extend *QPALMA* to take other or even multiple quality scores per nucleotide into account. For instance, for Roche's 454 sequencing platform, it can also be beneficial to extend the scoring model to appropriately model homo-polymer errors (e.g. by introducing additional states in the dynamic program for extending homo-polymers).

It appears interesting to include the downstream analysis of deriving the gene structure based on these reads into the pipeline and to estimate its error as well. This will be particularly interesting for predicting gene structures with alternative transcripts.

ACKNOWLEDGEMENT

We thank Georg Zeller, Richard Clark and Cheng Soon Ong for helpful comments and suggestions on this work. Moreover, we thank Detlef Weigel, Richard Clark and Christa Lanz for providing the Illumina Genome Analyzer reads for the *A.thaliana* genome. Korbinian Schneeberger is supported by DFG-funded ERA-PG Project *ARelatives*.

Conflict of Interest: none declared.

REFERENCES

Abouelhoda,M. *et al*. (2002) The enhanced suffix array and its applications to genome analysis. In *Proceedings of the Second Workshop on Algorithms in Bioinformatics*, Vol. 2452 of *Lecture Notes in Computer Science*. Springer, Berlin, Heidelberg, pp. 449–463.

Altun,Y. *et al*. (2003) Hidden Markov support vector machines. In *Proceedings of the 20th International Conference on Machine Learning*, Washington, DC, August 21–24, pp. 3–10.

Cortes,C. and Vapnik,V. (1995) Support vector networks. *Machine Learning*, **20**, 273–297.

Durbin,R. *et al*. (1998) *Biological Sequence Analysis: Probabilistic Models of Protein and Nucleic Acids*. 7th edn. Cambridge University Press, Cambridge.

Florea,L. *et al*. (1998) A computer program for aligning a cdna sequence with a genomic DNA sequence. *Genome Res.*, **8**, 967–974.

Freund,Y. and Schapire,R. (1997) A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, **55**, 119–139.

Gelfand,M. *et al*. (1996) Gene recognition via spliced sequence alignment. *Proc. Natl. Acad. Sci.*, **93**, 9061–9066.

Hillier,L. W. *et al*. (2008) Whole-genome sequencing and variant discovery in c. elegans. *Nat. Methods*, **5**, 183–188.

Kececioglu,J. and Kim,E. (2006) Simple and fast inverse alignment. In *Research in Computational Molecular Biology*, LNBI 3909, Springer, Berlin, Heidelberg, pp. 441–455.

Kent,W.J. (2002) BLAT–the BLAST-like alignment tool. *Genome Res.*, **12**, 656–664.

Li,M. *et al*. (2004) Adjust quality scores from alignment and improve sequencing accuracy. *Nucleic Acids Res.*, **32**, 5183–5191.

Meir,R. and Rätsch,G. (2003) An introduction to boosting and leveraging. In Mendelson,S. and Smola,A. (eds.) *Advanced Lectures on Machine Learning*, LNCS, Springer, Berlin, Heidelberg, pp. 119–184.

Mott,R. (1998) Trace alignment and some of its applications. *Bioinformatics*, **14**, 92–97.

Müller,K.-R. *et al*. (2001) An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, **12**, 181–201.

Rätsch,G. *et al*. (2006) Learning interpretable svms for biological sequence classification. *BMC Bioinformatics*, **7**(Suppl. 1), S9.

Schölkopf,B. and Smola,A.J. (2002) *Learning with Kernels*. MIT Press, Cambridge, MA.

Schulze,U. *et al*. (2007) Palma: mRNA to genome alignments using large margin algorithms. *Bioinformatics*, **23**, 1892–1900.

Slater,G. and Birney,E. (2005) Automated generation of heuristics for biological sequence comparison. *BMC Bioinformatics*, **6**(31).

Smith,T. and Waterman,M. (1981) Identification of common molecular subsequences. *Journal of Molecular Biology*, **147**(1), 195–197.

Sonnenburg,S. *et al*. (2002) New methods for splice-site recognition. In *Proceedings of the International Conference on Artificial Neural Networks*, LNCS vol. 2415, pp. 329–336.

Sonnenburg,S. *et al*. (2007) Accurate splice site prediction using support vector machines. *BMC Bioinformatics*, **8**(Suppl.10), S7.

Sundquist,A. *et al*. (2007) Whole-genome sequencing and assembly with high-throughput, short-read technologies. *PLoS One*, **2**, e484.

Usuka,J. *et al*. (2000) Optimal spliced alignment of homologous cdna to a genomic DNA template. *Bioinformatics*, **16**, 203–211.

Vapnik,V. (1995) *The Nature of Statistical Learning Theory*. Springer Verlag, New York.

Wold,B. and Myers,R.M. (2008) Sequence census methods for functional genomics. *Nat. Methods*, **5**, 19–21.

Zerbino,D. and Birney,E. (2008) Velvet: algorithms for de novo short read assembly using de bruijn graphs. *Genome Res* **18**, 821–829.

Zhang,M. and Gish,W. (2006) Improved spliced alignment from an information theoretic approach. *Bioinformatics*, **22**, 13–20.