SeeSite: Characterizing Relationships Between Splice Junctions and Splicing Enhancers

Christine Lo, Boyko Kakaradov, Daniel Lokshtanov and Christina Boucher

Abstract—RNA splicing is a cellular process driven by the interaction between numerous regulatory sequences and binding sites, however, such interactions have been primarily explored by laboratory methods since computational tools largely ignore the relationship between different splicing elements. Current computational methods identify either splice sites or other regulatory sequences, such as enhancers and silencers. We present an novel approach for characterizing co-occurring relationships between splice site motifs and splicing enhancers. Our approach relies on an efficient algorithm for approximately solving *Consensus Sequence with Outliers*, an NP-complete string clustering problem. In particular, we give an algorithm for this problem that outputs near-optimal solutions in polynomial time. To our knowledge, this is the first formulation and computational attempt for detecting co-occurring sequence elements in RNA sequence data. Further, we demonstrate that SeeSite is capable of showing that certain ESEs are preferentially associated with weaker splice sites, and that there exists a co-occurrence relationship with splice site motifs.

Index Terms—RNA splicing, exon splicing enhansers, randomized algorithms, PTAS, EPTAS

1 Introduction

RNA-splicing is the process of removing introns from pre-mRNA and merging the remaining exons into mRNA. During RNA-splicing, the spliceosome determines the location of the exons to merge by recognizing a motif at the *splice sites*, or the exon-intron boundaries where splicing occurs. Unfortunately, splice sites may be very degenerate, meaning that the corresponding sequence deviates dramatically from the motif [9]. When evolutionary changes weaken a splice site, regulatory elements such as ESEs (exonic splicing enhancers) and ISEs (intronic splicing enhancers) can be used to compensate for the degeneracy [3], [20], [34]. These regulatory elements are also thought to play a role in alternative splicing, as many alternative splice sites are weaker than constitutive sites [28], [39].

Current computational approaches for the analysis of splicing elements can be split into two categories based on their aim: those that detect splice sites, and those that detect splicing enhancers and silencers. Splice site identification tools find the boundaries between the exons and introns by using either RNA-seq data [30], [31], [13], [2] or previous annotations [22], [11]. While splice site identification is a fundamental problem, it only serves to partially solve the long-term goal of fully understanding how splicing activity is regulated since proximal sequences around these sites significantly affect splicing [9]. The second group of computational tools focus on identifying other splicing elements, such as enhancers and silencers that can occur in the introns (e.g. ISEs and ISSs) and exons (e.g. ESEs and ESSs). These tools search for motifs in a wide region around the splice sites but limit the search to exact matches of a motif. More specifically, they search for ℓ-mers that are statistically enriched in a set of cases versus controls [8], [15], [16], [25], [38]. For example, RESCUE-ESE [15] is a well-known program that compares exons with weak splice sites to those with strong splice sites reasoning that those with weak splice sites will have more enhancers. Both groups of computational tools focus on finding either splice sites or their regulatory elements and ignore the co-occuring relationship between the two. However, a co-occuring relationship between specific enhancer motifs and specific splice site motifs are known to exist [34], and an investigation of splicing activity necessitates a method that deciphers this co-occuring relationship.

Here we address the problem of identifying the co-occuring relationship between splice sites and splicing enhancers. We introduce a computational tool called *SeeSite*[1] that classifies splice sites based on their strength and uses this classification to identify

- *C. Lo and D. Lokshtanov are with the Computer Science and Engineering, University of California, San Diego, La Jolla, CA 92093, USA. C. Lo is a correponding author. E-mail: cylo@cs.ucsd.edu*
- *B. Kakaradov is with the Bioinformatics Graduate Program, University of California, San Diego, La Jolla, CA 92093, USA*
- *C. Boucher is with the Department of Computer Science, Colorado State University, Fort Collins, CO 80523-1873, USA. C. Boucher is a correponding author. E-mail: cboucher@cs.colostate.edu.*

1. A Java implementation of the method demonstrated in this paper is available at: http://bix.ucsd.edu/SeeSite.

co-occuring splicing enhancers in neighboring exon regions. SeeSite involves two distinct stages. In the first stage, SeeSite groups similar splice sites together and characterizes (i.e. finds the corresponding motif recognized by the spliceosome) and classifies each splice site as "weak" or "strong". In the second stage, SeeSite looks for enhancers in the exonic region of the weak splice sites of the motif (i.e. it locates and characterizes the ESEs).

The two stages of SeeSite both rely on characterizing a group of sequences; however the two stages have their differences. For instance, in the first stage, the location of the splice site sequences are given, whereas the location of the enhancer sequences need to be determined in the second stage. Also, the first stage requires the classification of sequences into "weak" and "strong", while no classification is needed in the second stage. Despite their difference, both stages of SeeSite use the same three-step algorithm which is general enough so that the parameters can be tailored to handle each stage appropriately. Figure 1 illustrates the flow of data among these two stages.

The three-step algorithm used in both stages of SeeSite involves building a graphical representation of the input data, finding all dense subgraphs, and characterizing the sequences in each dense subgraph. The first two steps of the algorithm can be handled using standard methods. However the third step requires solving a string clustering problem in the presence of noise since a number of the splice sites can be highly degenerate and can confound the consensus sequence. We formalize this problem as the *Consensus Sequence with Outliers* problem. This combinatorial problem is NP-complete [6] so we have to settle for heuristic algorithms to solve it. On the other hand, we show that choosing the parameters to our algorithm cleverly yields strong guarantees on the quality of the output. Specifically, we show that our algorithm is an efficient *polynomial time approximation scheme* (PTAS) unless the noise completely overwhelms the signal. We extend our theoretical findings by also giving a PTAS for *Consensus Sequence with Outliers* without any restrictions on the input.

A problem that is related to our model of detection of co-occuring splicing elements is motif-recognition, which models the biological challenge of finding transcription factor binding sites in genomic data [23]. Although, there are numerous programs that solve specific instances of this problem, including PROJECTION [7], Winnower [23], MITRA [14], MCL-WMR [4], and VAS [10], they are optimized to find the set of sequences that minimizes the pairwise distances. Thus, such programs are inappropriate for co-occuring splicing elements since they will not return sets of sequences that contain highly-degenerate (i.e. "weak") sites even though they maybe existing in the data. SeeSite overcomes this challenge, making it appropriate for detection of co-occuring splicing elements.

The rest of the paper is organized as follows: In Section 2, we give the formal definition for the *Consensus Sequence with Outliers* problem, algorithms to solve this problem, and the theoretical guarantees of these algorithms. In Section 3, we discuss the three-step algorithm and explicitly describe the flow of data between the first and second stage of SeeSite. In Section 4, we demonstrate the applicability of SeeSite in detecting co-occuring relationships between splice sites with specific ESEs.

2 Problem Definition and Algorithmic Approach

In this section, we focus on the formal definition of *Consensus Sequence with Outliers* and prove some properties about this problem, including the existence of an algorithm that produces a near-optimal solution in polynomial time.

2.1 The Consensus Sequences with Outliers Problem

Given a set of possible ℓ-mers, we would like to determine the consensus sequence, and the subset of ℓ-mers that are the most degenerate. The following problem defines this task.

Definition 1: We denote $d(x, y)$ to be the Hamming distance between the length-ℓ sequences x and y. Given n length-ℓ sequences $S = \{s_1, \ldots, s_n\}$ over a finite alphabet Σ and nonnegative integer k, the aim of the *Consensus Sequence with Outliers* problem is to find a consensus sequence, s, and subset, $S^* \subset S$, where $n - |S^*| = k$ and $\sum_{\forall t \in S^*} d(t, s)$ is minimal.
The problem is NP-hard [6], however, it is amenable to efficient approximation algorithms that are able to work well in practice.

We begin with some preliminary definitions before giving our algorithmic results. For a set S of length-ℓ sequences, we denote the consensus sequence of S as $c(S)$ and define it to be equal to the sequence that is obtained by picking a most-frequent character at every position with ties broken arbitrarily. We note that the tie-breaking will not affect our arguments. We denote the sum Hamming distance between a single sequence s and a set of sequences S as $d(S, s) = \sum_{\forall t \in S} d(t, s)$. The *Consensus Sequence With Outliers* problem can now be succinctly stated as follows: given a set S of sequences and integer k, the objective is to find a subset $S^* \subseteq S$ of size $n^* = n - k$ such that $d(S^*, c(S^*))$ is minimized. Observe that the consensus sequence of set S, $c(S)$, minimizes $d(S, c(S))$—that is, no other sequence x is closer to S than $c(S)$ but some $x \neq c(S)$ could achieve $d(S, x) = d(S, c(S))$

Given a subset $S^* \subseteq S$ we can compute $c(S^*)$ in polynomial-time. If we are given $c(S^*)$ for the optimal solution S^* (but not given S^* itself) then we can

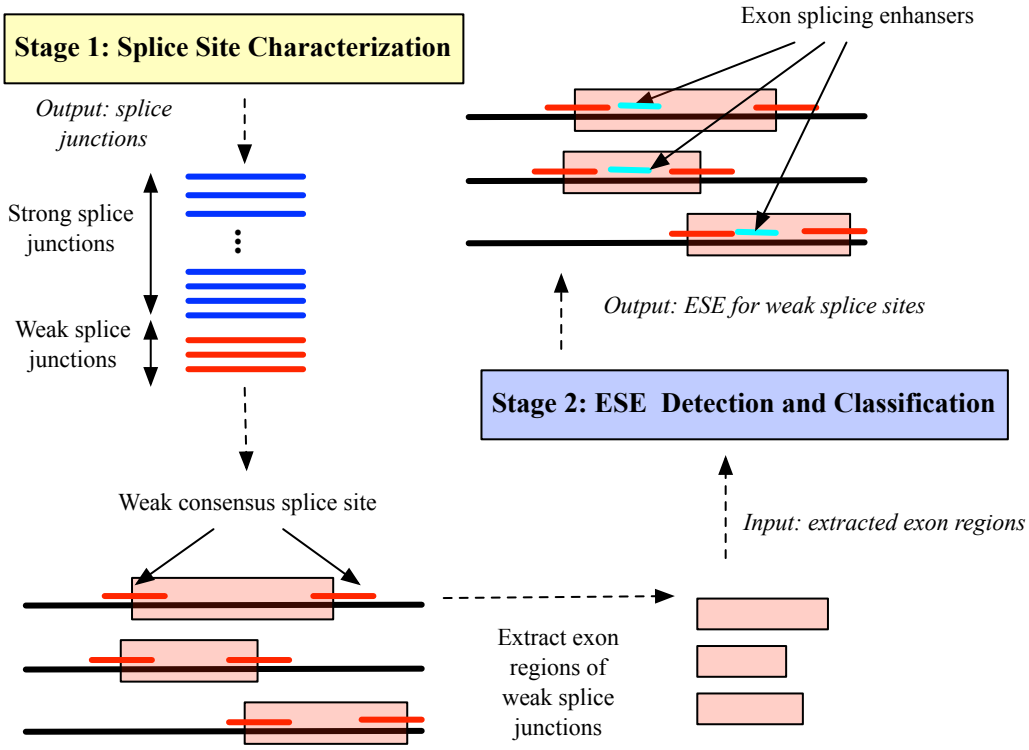

Fig. 1: The goal of SeeSite is the identification of co-occurring splice site-ESE motifs in two stages. First, splice site motifs are found based on a graph representation of ℓ-mers at splice site locations. In the second stage, the exon regions for each of the weak splice sites cited are examined to determine co-occurring ESEs. The output of the first stage determines the input to the second stage. The output of SeeSite is a set of splice sites classified by their canonical form, the identification of weak splice sites, and the group of co-occurring ESEs for each splice site motif.

recover S^* from $c(S^*)$ and S in polynomial-time since S^* is the set of the $n-k$ sequences in S that are closest to $c(S^*)$. Similarly, given any sequence x, we denote S_x as the subset of S containing the n^* sequences closest to x. By construction S_x satisfies the following inequality: $d(S', x) \geq d(S_x, x) \geq d(S_x, c(S_x))$ for any subset $S' \subseteq S$ of size n^*.

We give a heuristic algorithm for solving the *Consensus Sequence with Outliers* problem based on random sampling. The algorithm has two parameters r and t. It picks r sequences $S' = (s'_1, s'_2, ...s'_r)$ from S uniformly at random (with replacement), and finds the consensus sequence corresponding to S'. It repeats this process t times and outputs the best consensus sequence found. The pseudocode for this algorithm is given in Algorithm 1.

Algorithm 1

Input: S, k, r, t.
Output: a sequence s and subset, $S^* \subseteq S$ of size $n-k$.
Step 1: Initialize $s^* \leftarrow \emptyset$ and $d_{min} \leftarrow \infty$.
Step 2: Try t times:
(a) Choose a random subset of S of size r, denoted by S'.
(b) Order the sequences in S by increasing distance from $c(S')$ and let S_{max} be the last k sequences in this ordered set.
(c) Let S^* be equal to S/S_{max}.
(d) If $d(S^*, c(S^*))$ is less than d_{min} then update d_{min} and s^*.
Step 3: Return s^* and the corresponding S^*.

2.2 Approximation Guarantees

We prove guarantees on the quality of the solution output by Algorithm 1, when the parameters r and t are chosen appropriately. In particular we show that Algorithm 1 is an *efficient polynomial time approximation scheme* (EPTAS) for *Consensus Sequence with Outliers* if the data does not consist mainly of outliers. A *polynomial time approximation scheme* (PTAS) is a polynomial-time algorithm that outputs a $(1+\epsilon)$-approximate solution for every $\epsilon > 0$. Typically the running time upper bound, while polynomial for every *fixed* value of ϵ, grows very rapidly as ϵ tends to 0. If the exponent of the polynomial in the running time of the algorithm is independent of ϵ then the PTAS is said to be an *efficient PTAS* (EPTAS). To prove our bounds we prove the following technical

lemma, which states that if the sample S' was taken from an (unknown) optimal solution S^*, rather than from the entire input set S, then in expectation $c(S')$ is almost as good as the consensus sequence for the set S^*.

Lemma 1: For all $\epsilon > 0$ and σ, there exists a value of r such that the following holds: if S is a set of n length-ℓ sequences over the alphabet Σ, where the size of Σ is equal to σ, and S' is a subset of S of size r, $(s'_1, s'_2, ...s'_r)$, chosen uniformly at random, then $E[d(S, c(S'))] \leq (1+\epsilon)d(S, c(S))$.

Proof: We prove that there exists r such that $E[d(S, c(S'))] \leq (1+2\epsilon)d(S, c(S))$. Applying this weaker inequality with $\epsilon' = \epsilon/2$ then proves the statement of the Lemma. We assume, without loss of generality, that $c(S)$ is equal to 0^ℓ, $\epsilon \leq 1/16$, and $r \geq 8$. We restrict interest to column i of S, where $0 \leq i \leq \ell$, let d_i be the number of nonzero symbols in column i and let $z_i = n - d_i$. Observe that $d(S, c(S'))$ is equal to the sum over i of the number of sequences $s \in S$ such that $s[i] \neq c(S')[i]$. By linearity of expectation it is sufficient to prove that for every i we have $E[d(S[i], c(S')[i])] \leq (1+2\epsilon)d_i$.

First, we assume d_i is at most ϵn. Let q be the probability that $c(S')[i] \neq 0$. It follows that $E[d(S[i], c(S')[i])]$ is at most $d_i(1-q)+qn$. We determine an upper bound on the probability q:

$$\begin{aligned} q &\leq \sum_{x=\lceil r/2 \rceil}^{r} \binom{r}{x} (d_i/n)^x (1-d_i/n)^{r-x} \\ &\leq \sum_{x=\lceil r/2 \rceil}^{r} 2^r (d_i/n)^x \\ &\leq 2^r (d_i/n)^{\lceil r/2 \rceil} \frac{1-(d_i/n)^{\lceil r/2 \rceil}}{1-(d_i/n)}. \end{aligned}$$

Since $d_i/n \leq \epsilon \leq 1/16$, we get:

$$\begin{aligned} q &\leq 2^{r+1} (d_i/n)^{\lceil r/2 \rceil} \leq 2^{r+1} \epsilon^{\lceil r/4 \rceil} (d_i/n)^{\lfloor r/4 \rfloor} \\ &\leq 2^r \left(\frac{1}{16}\right)^{\lceil r/4 \rceil} \cdot 2 (d_i/n)^{\lfloor r/4 \rfloor} = 2 (d_i/n)^{\lfloor r/4 \rfloor}. \end{aligned}$$

It follows from the last inequality, and that $r \geq 8$, that $q \leq 2(d_i/n)^2$. Hence, we obtain the following bound on $E[d(S[i], c(S')[i])]$:

$$\begin{aligned} E[d(S[i], c(S')[i])] &\leq d_i(1-q) + qn \\ &\leq d_i + 2\left(\frac{d_i}{n}\right)^2 n \\ &\leq (1+2\epsilon)d_i \end{aligned}$$

Next, we assume that $d_i > \epsilon n$. We say that a symbol $\alpha \in \Sigma$ is a *good* symbol if there are at least $z_i - n\epsilon^2$ sequences in S that have the symbol α at column i; any symbol that is not good is *bad*. If $c(S')[i]$ is a good symbol then $d(S[i], c(S')[i])$ is at most $d_i + n\epsilon^2$ and hence, is at most $(1+\epsilon)d_i$ since $d_i > \epsilon n$. Let p be the probability that $c(S')[i]$ is a bad symbol then, $E[d(S[i], c(S')[i])]$ is upper bounded by $(1-p)(1+\epsilon)d_i + pn$. Lastly, we determine an upper bound on p to complete the proof.

Let α be a bad symbol and p_α be the probability that $c(S')[i]$ is equal to α. We note that in order for $c(S')[i]$ to be α, there has to be more positions equal to α than 0 in $S'[i]$. Let X be the difference between the number of positions equal to α and the number of positions equal to 0 in $S'[i]$. It follows that $p_\alpha \leq \Pr[X \geq 0]$. Let X_j be an indicator variable which is 1 if $s'_j[i]$ is equal to α, -1 if it is equal to 0, and 0 otherwise. Since α is a bad symbol, there are at least ϵ^2 more positions equal to 0 than positions equal to α in $S'[i]$ and therefore, $E[X_j] = \Pr[s'_j[i] = 0] - \Pr[s'_j[i] = \alpha] \leq -\epsilon^2$. By linearity of expectation, we obtain $E[X] = \Sigma_{j=1}^{r} E[X_j] \leq -r\epsilon^2$. Using this inequality, we get $\Pr[X \geq 0] \leq \Pr[X - E[X] \geq r\epsilon^2]$. Since the X_j variables are independent and the difference between the upper and lower bound of X_j is 2, we can use Hoeffding's inequality to obtain the following bound.

$$\Pr[X - E[X] \geq r\epsilon^2] \leq \exp\left(\frac{-2r^2\epsilon^4}{r2^2}\right) = \exp\left(\frac{r\epsilon^4}{2}\right)$$

By choosing $r = \min\left(n, \max\left(\frac{2\ln(\frac{\sigma}{\epsilon^2})}{\epsilon^4}, 8\right)\right)$, we get $p_\alpha \leq \frac{\epsilon^2}{\sigma}$. Finally, we bound p as follows: $p \leq \sum_\alpha p_\alpha \leq \sigma\frac{\epsilon^2}{\sigma} = \epsilon^2$. We now use this bound on p and our assumption that $d_i > \epsilon n$ to bound :

$$\begin{aligned} E[d(S[i], c(S')[i])] &\leq (1-p)(1+\epsilon)d_i + pn \\ &\leq (1+\epsilon)d_i + \epsilon^2 n \leq (1+2\epsilon)d_i \end{aligned}$$

□

If the number of outliers is small, then with reasonably high probability a small random subset of the input sequences will not contain any outliers. If the random sample does not contain outliers we can use Lemma 1 to tie the quality of the output solution with the quality of the optimum solution. Based on this intuition we can prove the following theorem.

Theorem 1: There exists a randomized EPTAS for *Consensus Sequence with Outliers* for inputs when $k \leq cn$ for $c < 1$. The algorithm runs in time $\frac{1}{(1-c)^r} \cdot f(\epsilon)(n\ell)^{O(1)}$ and outputs a $(1+\epsilon)$-approximate solution with probability $1/2$.

Proof: The algorithm selects a value for r such that for a random subset S' of the unknown optimal solution S^* the inequality $E[d(S^*, c(S'))] \leq (1+\frac{\epsilon}{3})d(S^*, c(S^*))$ holds. It follows from Lemma 1 that this can be done so that r only depends on ϵ. We show that a single iteration of the outer loop of Algorithm 1 with this choice for r yields a $(1+\epsilon)$-approximate solution with probability $(1-c)^r \cdot f(\epsilon)$.

Then setting $t = O\left(\frac{1}{(1-c)^r \cdot f(\epsilon)}\right)$ yields the statement of the theorem.

It remains to find a sufficient lower bound of the probability that the set returned by a single iteration of the outer loop of Algorithm 1 is a $(1+\epsilon)$-approximation. Since $k \leq cn$, it follows that the probability that S' is taken from an (unknown) optimal solution S^* is at least $(\frac{n-cn}{n})^r = (1-c)^r$. If S' is taken from S^* then by Lemma 1 we have that $E[d(S^*, c(S'))] \leq (1+\frac{\epsilon}{3})d(S^*, c(S^*))$. By Markov's inequality [18, p. 311] the probability that $d(S^*, c(S'))$ exceeds expectation by a factor at least $1+\frac{\epsilon}{3}$ is at most $\frac{1}{1+\frac{\epsilon}{3}}$. Hence, with probability $f(\epsilon)$ for some function f of ϵ we have that:

$$d(S^*, c(S')) \leq \left(1+\frac{\epsilon}{3}\right) d(S^*, c(S^*)) \cdot \left(1+\frac{\epsilon}{3}\right),$$

which is at most $(1+\epsilon)d(S^*, c(S^*))$ when $\left(\frac{\epsilon}{3}\right)^2 \leq \frac{\epsilon}{3}$. In particular, this holds if $\epsilon \leq 3$, concluding the proof. □

We note that one would expect natural inputs to contain substantially fewer outliers than $n/2$, and that Markov's inequality is a very pessimistic bound for the probability of achieving expectation. It is likely that for reasonable inputs Algorithm 1 performs much better in practice than the proved bounds. In fact, on our synthetic data the algorithm vastly outperformed the theoretical bounds.

Using Lemma 1 we can also get a simple deterministic PTAS (but not an EPTAS) for *Consensus Sequence with Outliers* without any assumptions on the relationship between k and n. Specifically we prove the following theorem.

Theorem 2: There exists a PTAS for *Consensus Sequence with Outliers*.

Proof: It follows from Lemma 1 that there exists an integer r such that if S', the set of r of sequences chosen from S, is from an (unknown) optimal solution S^* then $E[d(S^*, c(S'))] \leq (1+\epsilon)d(S^*, c(S^*))$. Some subset S' of S^* must achieve the expectation. The algorithm guesses this set S' by trying all possible n^r subsets of S of size r. Let $x = c(S')$. The algorithm returns the set S_x of the n^* sequences closest to x. This set satisfies $d(S_x, c(S_x)) \leq d(S_x, x) \leq d(S^*, x) \leq (1+\epsilon)d(S^*, c(S^*))$, concluding the proof.

□

3 METHODS

There are two distinct stages of SeeSite: the first stage characterizes the splice sites, while the second stage detects enhancers in the exons corresponding to the weak splice sites. Both stages run the same three-step algorithm. In this section, we describe the three steps of the algorithm in detail, and how the two stages are pipelined together to create SeeSite. The input to SeeSite is the intron-exon or exon-intron boundary regions containing the splice junctions and their neighboring exons (details on how these sequences are obtained are described in the preprocessing section). The input parameters are: the minimum subgraph size (denoted as m), the maximum number of mismatches (denoted as d), and the maximum number of mismatches for the existence of an edge (denoted as b). There is an option to restrict the search to ESEs or splice junctions that have a specific canonical form. The output of SeeSite is all splice junctions, grouped by their consensus sequence, and a set of ESEs for each consensus sequence. By comparing the set of ESEs for differing consensus sequences, co-occurrence relationships become evident.

3.1 Preprocessing Step

The input to SeeSite is a set of sequences flanking the splice junctions, where the splice junctions are either known or putative. When the splice sites are known, these input sequences can be trivially identified by examining the sequences flanking the splice sites. For putative sites not in the reference transcriptome the flanking sequences need to be explicitly identified, which can be done using a short read alignment program that is capable of dealing with RNA-seq data (e.g. TopHat [30], OSA [19], GSNAP [33]).

3.2 Pipelining Together the Stages of SeeSite

In the first stage, SeeSite finds and characterizes all possible splice sites. These sites are grouped according to their consensus sequence, i.e. all splice sites with the same consensus sequence are in the same group, with some of the splice sites classified as being weak. As previously mentioned, a splice site will be classified as weak if the sequence marker corresponding to the splice site is highly degenerate, and has been determined to be an outlier sequence.

The exons are partitioned into smaller subgroups that constitute the input for the second stage as follows: all exons whose corresponding splice sites have the same consensus sequence are grouped together, then all exons whose splice site was classified as strong are removed. What remains are groups of exons whose splice junctions have the same consensus sequence and have been classified as being weak. The three-step algorithm is run on each (smaller) set of exon sequences in the second stage. The parameters that specify the amount of allowed degeneracy (i.e. parameters b and d) are dramatically reduced in the second stage since ESEs are inherently shorter and more well-conserved. Figure 1 illustrates the flow of data among the two stages of SeeSite

3.3 Three-Step Algorithm

The steps of the basic algorithm used for both stages of SeeSite are as follows: graph construction, dense

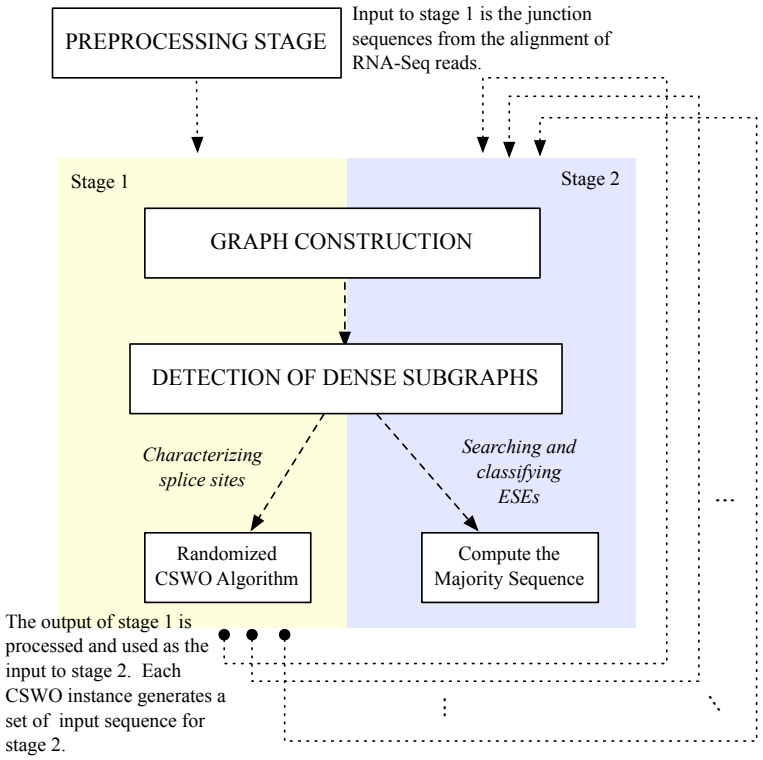

Fig. 2: The core algorithm is a three-step process that runs twice in succession. On the first run (yellow region marked Stage 1), the algorithm receives a set of proximal sequences from known or putative splice junctions and determines one or more motifs for their splice sites. On the second run (blue region marked Stage 2), the algorithm receives multiple subsets of sequences from the first run—each corresponding to a weak splice site motif—and searches their exonic portions for neighboring ESEs.

subgraph detection, and recovery of the splice junctions and ESEs. An overview of this algorithm is shown in Figure 2.

Step 1: Graph Construction

In our graphical representation of the data set, each occurring ℓ-mer is represented by a vertex and the construction of our graph ensures that ℓ-mers corresponding to the same motif sequence are represented by dense subgraphs (though the converse need not hold). Thus, the problem of detecting ESEs and splice sites is converted to finding dense subgraphs of size at least m in our constructed graph G. Here, we remind the reader that m is an input parameter, and n is the number of input sequences. We now give a formal definition of the constructed graph. Our definition makes the simplifying assumption that all input sequences have length at most L. We note that the definition can be trivially extended to the case where the input sequences have varying lengths.

1) The vertex set contains a vertex $v_{i,j}$ representing the l-length subsequence in sequence i starting at position j, for each i and $j = 1, 2, \ldots, L-l+1$. There are at most $n(L-l+1)$ vertices.
2) Each pair of vertices $v_{i,j}$ and $v_{i',j'}$, for $i \neq i'$ is joined by an edge when the Hamming distance between the two represented subsequences is at most b.

This graph is represented by a symmetric adjacency matrix, where each entry is 0 for a non-edge, and 1 for an edge. We reduce the running time of searching G by considering subgraphs of G, $\{G_0, G_1, \ldots, G_{L-1}\}$, where G_i is the subgraph induced by a reference vertex, denoted as $v_{R,i}$, and its neighbors (for some arbitrary choice of reference sequence R). Similar graph constructions have been used by Yang and Rajapakse [36], Pevzner and Sze [23], and Yang et al. [35].

Step 2: Detection of Dense Subgraphs

We implemented a modified version of the MCQ algorithm of Tomita and Seki [29] to enumerate all dense subgraphs of size at least m. Each dense subgraph represents a set of ℓ-mers, such that the Hamming distance between any pair of ℓ-mers in the set is at most b. We chose MCQ due to the experimental work showing that when compared with other existing algorithms, it is the most efficient and practical for dealing with large graphs. The underlying idea of this branch-and-bound, depth-first search method is to begin with a small dense subgraph, add a vertex to it if and only if it is connected to some minimum number of vertices already contained in the subgraph, and halt

when no other vertices can be added. If the subgraph has size at least m then it is returned. The vertex sets outputted by the MCQ algorithm correspond to sets of ℓ-mers that need to be considered further in the next stage of the algorithm.

Step 3: Consensus Sequence with Outliers Algorithm

Each set of ℓ-mers identified in the previous step is considered to be a *Consensus Sequence with Outliers* instance in this step. Algorithm 1 is used to solve each *Consensus Sequence with Outliers* instance. The parameters of the *Consensus Sequence with Outliers* instance is varied based on which stage of SeeSite is being run. When searching for splice sites in the first stage, the number of outliers is equal to $\lfloor m/4 \rfloor$, and when searching for ESEs in the second stage there are no outliers. Both the *Consensus Sequence with Outliers* problem definition and algorithmic methods are adaptive in that they are provably efficient, regardless of the number of outliers. In Subsection 2.2, we proved that Algorithm 1 runs efficiently even when the number of outliers are arbitrarily large or small. When there are no outlier sequences, Algorithm 1 simply returns the majority sequence and, therefore, is efficient.

4 Experimental Results and Discussion

Existing computational methods search for either splice sites or ESEs, but overlook the connection between the two groups of motifs. In this section, we demonstrate the ability of SeeSite to identify both enhancer sequences and splice site motifs that have an unusually strong tendency to co-occur.

4.1 Identification of Co-occurring Relationships in Genes with Known Splice Sites

The human genome has many multi-exon genes with excellent EST coverage and high-quality annotation, and thus, provides a good source of known splice sites on which we can evaluate the capability of SeeSite. Our benchmark dataset consists of 9,487 known splice sites from the human genome (GRCh37 assembly) and its reference annotation (RefSeq). For each known splice junction, we extracted two 100nt subsequences centered at the 5′ and 3′ splice sites flanking each known intron and provided them to SeeSite to search for proximal ESEs.

In the first stage of SeeSite, we grouped all splice sites according to their consensus sequence, and classified each site as being strong or weak. For this stage, we varied the input parameters ℓ, d and b, and only considered groups of splice junctions of size greater than $m = 100$. We used a metric referred to as the *consensus value* (CV), which ranges from 100 (perfect consensus) to 0 (worst consensus), to gauge the degeneracy (or strength) of the sequence corresponding to the splice junction [24], [38]. The CV guided our choice of k, the number of outlier sequences allowed. We observed that when k was greater than 25% of the size of the instance, the CV of some of the outlier sequences was greater than 60. Therefore, we set the value of k to be 25% of the size of the instance to ensure that we classify degenerate splice sites as "weak".

SeeSite characterized 9,308 of the 9,487 known splice sites, 87% of these sites overlapped with known gene models that have been verified by ESTs. Less than 2% of splice sites had been identified by ESTs but were not found by SeeSite. SeeSite characterized the GT-AG, GC-AG, and AT-AC splicing site patterns in 87%, 3%, and 0.5% of the 9,308 splice sites considered. The remaining 179 splice sites were too degenerate to be characterized; each of these sites contained at most 2nt consistent with one of the identified splice site motifs. For the splice sites characterized as GT-AG, 2% matched perfectly to the consensus (CV of 100), 4% had a CV between 90 and 80, 26% had a CV between 80 and 77, and the remaining 68% had a CV less than 77. All splice sites that were found to be outlier sequences had a CV less than 60.

In the second stage, SeeSite identified ESEs corresponding to the groups of weak splice sites. For each group of weak splice sites of the same consensus form, we ran the second stage with varied values of d and b, and ℓ equal to 5 and 6 and $k = 0$. Previous results [9], [15] demonstrating that commonly occurring ESEs were either 5-mers or 6-mers guided our choice of ℓ. We validated our findings by comparing the ESEs found by SeeSite with those identified by RESCUE-ESE [15][2]; 95% of the ESEs that were identified by SeeSite were also identified by RESCUE-ESE [15], and less than 1% of the ESEs that were identified by RESCUE-ESE were not identified SeeSite [15].

Table 1 gives all splice site forms characterized by SeeSite, and for each splice site form, the ESEs that occur most frequently. As witnessed in the table, the ESEs that occur most frequently with one splice site form occur infrequently with splice sites of other forms. Our results confirm the existence of co-occurring splicing elements, i.e. a pairing of splice sites with specific ESEs. For the weak splice sites of each form, GT-AG, GC-AC, and AT-AC, we determined the ESEs that occur most frequently and those that occur the most infrequently and compared these across different splicing sites. We witnessed that the occurrence of several ESEs have high probability of occurring with the corresponding splice site having a specific pattern (e.g. GT-AG, GC-AC, and AT-AC), while others have low probability of occurring with splice sites of other forms. There exists strong

2. The ESEs identified by RESCUE-ESE are found at: http://genes.mit.edu/burgelab/rescue-ese/ESE.txt.

evidence for a pairing of ESEs to splicing sites.

To address the conjecture that ESEs are more likely to be present when the splice junction is weak rather than strong, we ran the second stage of SeeSite on the strong splice sites, and compared the number of exons that are associated with strong splice sites and contain an ESE, with the number of exons that are associated with weak splice sites and contain an ESE. In this context, we refer to a splice site as strong if the CV is greater than or equal to 85. We found that 90% of weak splice sites are paired with an ESE, as opposed to 30% of strong splice sites. This statistic supports the idea that co-occurring pairs are contributing to splicing by compensating for a lack of strong splicing signals.

The results suggest the existence of several non-canonical splice site patterns and demonstrates a possible synergistic relationship between ESEs and different classes of splice site patterns. Validation experiments such as fluorescence-activated screens of splicing reporters [32] are needed to resolve whether the relationships detected by SeeSite are biologically significant. We believe that the introduction of SeeSite will inspire future investigations on co-occurring relationships between splice sites and other regulatory elements (i.e. exon silencers, intron silencers, intron enhancers).

4.2 Identification of Co-occurring Relationships in Genes with Putative Splice Sites

In addition, we evaluated SeeSite in a setting where the splice sites are unknown, and RNA-seq data is needed to extract these putative junctions not contained in the reference transcriptome. To identify their potential splice sites, we obtained single-ended 75nt RNA-Seq reads from the Brain tissue dataset of Illumina's Human BodyMap 2.0 project (GEO Accession number GSE30611) and aligned them to the human genome and transcriptome references using TopHat (with default settings) [30]. Then, we removed 6,629 known splice junctions that were expressed in our data, which represent about 70% of the 9,487 known splice sites described earlier. Finally, we used HMMsplicer (with default settings) [13] on the remaining alignments to identify 2,690 pairs of novel splice sites which represent both canonical and non-canonical junctions in the aligned reads. This preprocessing of RNA-seq data yields 100nt regions around putative splice junctions suitable for input to SeeSite.

Similar results were observed on the the putative splicing data, consisting of 2,690 non-reference splicing regions. SeeSite identified splice junctions in 2,600 (96%) of the 2,690 genes considered. The most common splice sites, GT-AG, GC-AG, and AT-AC, are found in 98.7%, 1.1% and 0.1% of human introns, respectively. This is not surprising since similar results have been reported on similar datasets by Dimon et al. [13], Au et al. [2], and Trapnell et al. [30]. There are also splice junctions that do not have GT-AG, GC-AG or AT-AC splice sites, however, this was so rare that we did not have a large enough set of weak splice junctions to determine co-occurring relationships. ESEs that frequently occured in genes containing a GT-AG splice junction included: GAGAAA, AAAAGA, AAGAAA; each of these ESEs occurred in less than 10% of the containing splice junctions of different forms. Similarly, the ESEs in the exons of the genes having splice junctions of the form GC-AG or AT-AC were frequently one of the enhancers in the following respective sets: {AAGCAG, CAGAAG} and {ATGGAA, AAAGCT}. The sets of frequently occurring ESEs for GT-AG, GC-AG and AT-AC were disjoint, meaning no ESEs that co-occurred with one splice site form also co-occurred with another splice site form.

4.3 Practical Considerations: Memory and Time

We evaluated the memory and time requirements of SeeSite. Its wall-time depends on the computing resources available to the user since it is a multithreaded application. For evaluation purposes we used four threads, a setting that is suitable for most servers. The memory requirements depend on a number of factors, including size of exons, and number of genes considered. SeeSite required a maximum of 32 GB and 8 hours to complete on our benchmark dataset consisting of 9,487 known splice sites from the human genome (GRCh37 assembly) (see Table 2).

5 Conclusion

To our knowledge, SeeSite is the first tool that directly identifies co-occurring relationships between splice sites and ESEs. While there are numerous programs that detect either splice sites or proximal enhancers and silencers, the classification of co-occurring relationships from a computational perspective has been more elusive until now. We believe that the introduction of SeeSite will inspire future investigations on co-occurring relationships between splice sites and other regulatory elements (i.e. intron silencers and exon silencers).

The results suggest the existence of several non-canonical splice site patterns and demonstrates a possible synergistic relationship between ESEs and different classes of splice site patterns. Validation experiments such as fluorescence-activated screens of splicing reporters [32] are needed to resolve whether the relationships detected by SeeSite are biologically significant or simply spurious correlations detected in the data. However, this is unlikely, given the strength and frequency of a number of the patterns. Determining the exact biological relationship of these paired splicing elements warrants further investigation.

Splicing Pattern	Frequently occurring ESEs	Infrequently occurring ESEs
GT(G/A)AGT‖(T/C)AG	GAGAAA (27%) AAAAGA (24%) AAGAAA (24 %)	AAGCAG (7%) CAGAAG (2%) ATGGAA (0%) CAGAAG (4%)
GCA(T/A)G(G/T)‖(T/A)AC	AAGCAG (35%) CAGAAG (34%)	GAGAAA (2%) AAAAGA (2%) ATGGAA (3%) AAAGCT (5%)
ATG(C/A)T(G/A)‖(G/T)AC	ATGGAA (41%) AAAGCT (31%)	AAGCAG (2%) CAGAAG (3%) AAAAGA (2%)

TABLE 1: Some examples that illustrate the co-occuring relationship between splice sites and ESEs that control splicing expression. We searched for all possible ESEs in over 3000 weak splice sites of the form GT-AG, over 500 weak splice sites of the form GC-AC, and over 100 weak splice sites of the form AT-AC. For each splice site pattern we listed the ESEs that frequently occur with splice junctions of the corresponding form and infrequently occur with splice junctions of other forms. The percentage of genes containing the corresponding ESE and weak splice site of the given form is given in brackets.

Genome	Number of splice sites	Time	Memory
Human genome (GRCh37 assembly)	9,487	477 min. (approx. 8 hours)	32 GB
Arabidopsis (Columbia Reference)	9,231	491 min. (approx. 8 hours and 11 min.)	36 GB

TABLE 2: Running time and memory usage of SeeSite on a select number of gene of of the human and Arabidopsis genomes.

ACKNOWLEDGEMENT

The authors would like to thank Asa Ben-Hur and Michael Hamilton from Colorado State University for many insightful comments and discussions. Funding was provided by National Institutes of Health (NIH), National Science Foundation (NSF), and Natural Sciences and Engineering Research Council of Canada (NSERC), Postdoctoral Fellowship Program. For the duration of this project, CL was funded by NSF (grant CCF-1115206) and the NSF Post-Graduate Fellowship Program, BK was funded by the NSF Post-Graduate Fellowship Program, DL was supported by the Bergen Research Foundation under the "Beating Hardness by Pre-processing" grant, and CB was funded by NIH (grant 3P41RR024851-02S1) and NSERC Post-Doctoral Fellowship Program.

REFERENCES

[1] M. D. Adams, J. M. Kelley, J. D. Gocayne, M. Dubnick, M. H. Polymeropoulos, H. Xiao, C. R. Merril, A. Wu, W. R. McCombie, J. C. Venter Complementary DNA Sequencing: Expressed Sequence Tags and Human Genome Project *Science* 252(5013): 1651-1656, 1991.

[2] K.F. Au, H. Jiang, L. Lin, Y. Xing, and W.H. Wong. Detection of splice junctions from paired-end RNA-seq data by SpliceMap. *Nucl. Acid. Res.*, 38(14):4570–4578, 2010.

[3] Y. Barash, J. A. Calarco, W. Gao, Q. Pan, X. Wang, O. Shai, B. J. Blencowe, B. J. Frey Deciphering the splicing code. *Nature* 465(7294): 53-59, 2010.

[4] Boucher, C., Church, P., and Brown, D. A graph clustering approach to weak motif recognition. *Proc WABI 2007*, pages 149–160.

[5] C. Boucher, G. Landau, A. Levy, D. Pritchard, and O. Weimann. On Approximating String Selection Problems with Outliers. In Proc of *CPM*, pages 427–438, 2012.

[6] C. Boucher, C. Lo, and D. Lokshtanov. Outlier Detection for DNA Fragment Assembly. arXiv:1111.0376.

[7] Buhler, J., and Tompa, M. Finding motifs using random projections. *J Comput Biol*, 9 (2):225–242, 2002.

[8] L. Cartegni, J. Wang, Z. Zhu, M.Q. Zhang, and A.R. Krainer. ESEfinder: A web resource to identify exonic splicing enhancers. *Nucl. Acid. Res.*, 31(13):3568–3571, 2003.

[9] L.A. Chasin. Searching for Splicing Motifs. In *Alternative Splicing in Postgenomics Era*. B. Blencowe and B. Graveley, eds., pp.. Landes Bioscience, Austin, TX 78701 85-106.

[10] Chin, Francis Y.L., and Leung, C.M. An efficient algorithm for string motif discovery. *Proc APBC 2006P*, pages 79-88.

[11] S. Degroeve, Y. Saeys, B. De Baets, P. Rouzé, and Y. Van de Peer. SpliceMachine: predicting splice sites from high-dimensional local context representations. *Bioinformatics* 21(8):1332-8, 2005.

[12] M. Deutsch, and M. Long. Intron-exon structures of eukaryotic model organisms. *Nucl. Acid. Res.*, 27(15):3219-28, 1999.

[13] M.T. Dimon, K. Sorber, and J.L. DeRisi. HMMSplicer: a tool for efficient and sensitive discovery of known and novel splice junctions in RNA-Seq data. *PLoS ONE*, 5(11):e13875, 2010.

[14] Eskin, E. and Pevzner, P.A. Finding composite regulatory patterns in DNA strings. *Bioinformatics*, **18(1)**:354–363, 2002.

[15] W.G. Fairbrother, R.F. Yeh, P.A. Sharp, and C.B. Burge. Predictive identification of exonic splicing enhancers in human genes *Science*, 297(5583):1007–1013, 2002.

[16] A. Fedorov, S. Saxonov, L. Fedorova, and I. Daizadeh. Comparison of intron-containing and intron-lacking human genes elucidates putative exonic splicing enhancers. *Nucl. Acid. Res.*, 29(7):1464–1469, 2011.

[17] M.G. Grabherr et al. Full-length transcriptome assembly from RNA-seq data without a reference genome. *Nat. Biotechnol.*, 29(7):644-52, 2011.

[18] F. Grimmett, and D. Stirzaker. Probability and random processes. *Oxford University Press*, 3 edition, 2001.

[19] J. Hu, H. Ge, M. Newman, and K. Liu. OSA: A fast and accurate alignment tool for RNA-Seq. *Bioinformatics*, 28(14): 1933-1934, 2012.

[20] A.L. Lear, L.P. Eperon, I.M. Wheatley, and I.C. Eperon. Hierarchy for 5′ splice site preference determined in vivo. *J. Mol. Biol.*, 211(1):103–15, 1990.

[21] A. Mortazavi, B. A. Williams, K. McCue, L. Schaeffer, and B. Wold Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nature Methods*, 5: 621 - 628, 2008.

[22] M. Pertea, X. Lin, and S.L. Salzberg. GeneSplicer : a new

computational method for splice site prediction. *Nucleic Acids Res* 29(5):1185-90, 2001.

[23] P. Pevzner and S.H. Sze. Combinatorial approaches to finding subtle signals in DNA sequences. In Proc. of *ISMB* 8:344–354, 2000.

[24] P. Senapathy, M.B. Shapiro, and N.L. Harris. Splice junctions, branch point sites, and exons: sequence statistics, identification, and applications to genome project. *Methods Enzymol.*, 183:252–78, 1990.

[25] M. Sironi et al. Silencer elements as possible inhibitors of pseudoexon splicing. *Nucl. Acid. Res.*, 32(5):1783–1791, 2004.

[26] S. Sonnenburg, G. Schweikert, P. Philips, J. Behr, and G. Rätsch. Accurate splice site prediction using support vector machines. *BMC Bioinformatics*, 8(Suppl 10):S7, 2007.

[27] S. Stamm et al. ASD: a bioinformatics resource on alternative splicing. *Nucl. Acid. Res.*, 34(1):D46–D55, 2005.

[28] T.A. Thanaraj, and S. Stamm. Prediction and statistical analysis of alternatively spliced exons. *Prog. Mol. Subcell. Biol.*, 31:1–31, 2003.

[29] E. Tomita, and T. Seki. An efficient branch-and-bound algorithm for finding a maximum clique. In Proc. of *DMTCS*, 278-289. 2003.

[30] C. Trapnell, L. Pachter, and S.L. Salzberg. TopHat: discovering splice junctions with RNA-Seq. *Bioinformatics*, 25(9):1105–1111, 2009.

[31] K. Wang et al. MapSplice: accurate mapping of RNA-seq reads for splice junction discovery. *Nucl. Acid. Res.*, 38(18):e178, 2010.

[32] Y. Wang, M. Ma, X. Xiao, Z. Wang Intronic splicing enhancers, cognate splicing factors and context-dependent regulation rules *Nature Struct. and Molec. Bio.*, 2012

[33] T.D. Wu and S. Nacu. Fast and SNP-tolerant detection of complex variants and splicing in short reads. *Bioinformatics*, 26 (7): 873-881, 2010.

[34] X. Xiao, Z. Wang, M. Jang, and C.B. Burge. Coevolutionary networks of splicing cis-regulatory elements. *Proc. Natl. Acad. Sci.* 104:18583-18588, 2007.

[35] X. Yang, K.S. Dorman, and S. Aluru. Reptile: Representative tiling for short read error correction. *Bioinformatics*, 26(20):2526–2533, 2010.

[36] X. Yang, and J. Rajapakse. Graphical approach to weak motif recognition. *Gen. Info.* **15(2)** (2004) 52–62.

[37] G. Yeo and C.B. Burge Maximum Entropy Modeling of Short Sequence Motifs with Applications to RNA Splicing Signals *J. of Comp. Bio.*, 11(2-3): 377-394, 2004.

[38] X.H. Zhang, C.S. Leslie, and L.A. Chasin. Computational searches for splicing signals. *Methods*, 37(4):292–305, 2005.

[39] C.L. Zheng, X.D. Fu, and M. Gribskov. Characteristics and regulatory elements defining constitutive splicing and different modes of alternative splicing in human and mouse. *RNA* 11(12):1777–1787, 2005.