

Sequence analysis

## Detection of generic spaced motifs using submotif pattern mining

Edward Wijaya<sup>1,2</sup>, Kanagasabai Rajaraman<sup>1</sup>, Siu-Ming Yiu<sup>4</sup> and Wing-Kin Sung<sup>2,3,\*</sup>

<sup>1</sup>Institute for Infocomm Research, 21 Heng Mui Keng Terrace, Singapore 119613, <sup>2</sup>School of Computing, National University of Singapore, Singapore 119260, <sup>3</sup>Genome Institute of Singapore, 60 Biopolis Street, #02-01 Genome, Singapore 138672 and <sup>4</sup>Department of Computer Science, The University of Hong Kong, Pokfulam Road, Hong Kong

Received on December 5, 2006; revised and accepted on March 16, 2007

Advance Access publication May 5, 2007

Associate Editor: John Quackenbush

### ABSTRACT

**Motivation:** Identification of motifs is one of the critical stages in studying the regulatory interactions of genes. Motifs can have complicated patterns. In particular, spaced motifs, an important class of motifs, consist of several short segments separated by spacers of different lengths. Locating spaced motifs is not trivial. Existing motif-finding algorithms are either designed for monad motifs (short contiguous patterns with some mismatches) or have assumptions on the spacer lengths or can only handle at most two segments. An effective motif finder for generic spaced motifs is highly desirable.

**Results:** This article proposes a novel approach for identifying spaced motifs with any number of spacers of different lengths. We introduce the notion of *submotifs* to capture the segments in the spaced motif and formulate the motif-finding problem as a frequent submotif mining problem. We provide an algorithm called SPACE to solve the problem. Based on experiments on real biological datasets, synthetic datasets and the motif assessment benchmarks by Tompa *et al.*, we show that our algorithm performs better than existing tools for spaced motifs with improvements in both sensitivity and specificity and for monads, SPACE performs as good as other tools.

**Availability:** The source code is available upon request from the authors.

**Contact:** [ksung@comp.nus.edu.sg](mailto:ksung@comp.nus.edu.sg)

**Supplementary information:** Supplementary data are available at *Bioinformatics* online.

### 1 INTRODUCTION

One of the major challenges facing biologists today is understanding the regulatory mechanism of genes. This challenge includes detection of transcription factor binding sites involved in regulation and discovery of regulatory networks. The problem of *de novo* identification of transcription factor binding site motifs has been widely studied and a number of motif-finding algorithms have been proposed under categories such as profile-based methods e.g. Gibbs sampler (Lawrence *et al.*, 1993), MotifSampler (Thijs *et al.*,

2002), SeSiMCMC (Favorov *et al.*, 2005), GAME (Wei and Jensen, 2006), Improbizer (Ao *et al.*, 2004), consensus-based methods e.g. Weeder (Pavesi *et al.*, 2001), MITRA (Eskin and Pevzner 2002), Gemoda (Jensen *et al.*, 2006) and hybrid methods that use a combination of the above two methods, e.g. (Hertz and Stormo, 1999).

However, motif-finding continues to be a difficult problem. Recently Tompa (Tompa *et al.*, 2005) conducted an assessment of 13 popular motif discovery algorithms over 56 datasets drawn from *Homo sapiens*, *Mus musculus*, *Drosophila melanogaster* and *Saccharomyces cerevisiae* genomes, and found that all the algorithms performed unimpressively overall (barring yeast datasets).

One reason is that motifs can have complicated patterns. As pointed out by Eisen in a recent survey (Eisen, 2005), regulatory motifs could be highly complex in the biological context. Many motifs are known to be composite patterns which are groups of monad patterns (short contiguous patterns with some mismatches) that occur relatively near each other (Harbison *et al.*, 2004). For example, the binding site for *ArcA-P*, a transcription factor for regulating gene related to the respiratory metabolism in *Escherichia coli* (Liu and DeWulf, 2004), can be regarded as two conserved segments, separated by a spacer of length approximately 6 (McGuire *et al.*, 1999). Another example is *Mcm1* (Kato *et al.*, 2004) or often called as the early cell cycle box (ECB) (Tavazoie *et al.*, 1999) which has three segments and two spacers. Note that a spacer does not necessarily mean that the characters in the spacer are completely random and arbitrary, but these characters are not very conserved in different instances.

In fact, in some regulatory mechanisms, a single transcription factor may bind to two or more sites that are relatively close to each other—as is frequently the case, for instance, of RNA polymerase (Record *et al.*, 1996). Identifying these sites is similar to finding a spaced motif. Spaced motifs may also be associated with co-regulated genes that share two or more transcription factors and the binding sites are often recognized by different macromolecular complexes that make contact with one another (Owen and Zelent., 2000; Werner, 1999). Our focus in this article is to find such complex motifs that could contain spacers.

\*To whom correspondence should be addressed.

Most of the existing algorithms are mainly designed for monad motifs. Applying these algorithms to locate spaced motifs may not be effective. By treating a spaced motif as a single monad pattern, the motif instances may not be very similar, i.e. the signal may not be strong to be detected, due to the many random (non-conserved) characters in the spacers. Or if we try to locate the individual segments of a spaced motif using these algorithms, some of the segments may be too short and may not be easily detected.

On the other hand, there are algorithms designed for spaced motifs. The methods used by existing algorithms can be classified into the following approaches. The first and the most common approach is to assume that all the spacers in the same motif are all of the same fixed length [e.g. SesiMCMC (Favorov *et al.*, 2005), OligoDyad (van Helden *et al.*, 2000)]. However, in real cases, this is not the case. Another approach to handle spacers is to enumerate all possible spacer lengths between two composite segments (e.g. YMF developed in (Sinha and Tompa, 2000) and BioProspector (Liu *et al.*, 2001)). Although this approach can find motifs with spacers of varying length, it is inherently inefficient and is difficult to extend to more than two segments. And it may not be practical for long motifs. The third approach to locate spaced motifs is to find the monad segments first (e.g. MITRA in (Eskin and Pevzner, 2002)), then based on the locations of monad segments, locate a set of possible dyads (spaced motifs with two segments). The algorithm of MITRA relies on a specially designed data structure (mismatch tree data structure) to quickly identify possible monad segments. There are other methods [e.g. (Carvalho *et al.*, 2005; Marsan and Sagot, 2000)] that make use of data structures such as suffix tree to store the regularly spaced motif before finally identifying the motif pairs to speed up the process. Almost all existing approaches only handle spaced motifs with two segments.

In this article we propose a new approach for finding spaced motifs, and develop a novel motif-finding algorithm that offers flexibility in handling spacers with different lengths, the number of segments and variations in segment lengths. We formulate the motif finding problem as a frequent itemset mining and present an algorithm called SPACE for finding these motifs. Experimental results show that the approach is promising.

Our approach is similar to TEIRESIAS (Rigoutsos and Floratos, 1998) in building longer motif from shorter blocks. Yet, TEIRESIAS is computationally expensive. It uses a *convolution* strategy to stitch the shorter blocks exhaustively to find maximal patterns. It also does not handle mismatches. On the other hand, our novel approach provide further advantages. We allow flexibilities in terms of allowing mismatches and provide an efficient method to find the pattern.

## 2 SPACED MOTIFS AND THE SUBMOTIFS

In this section, we provide the formal definition of a generic spaced motif and discuss the notion of *submotif* which is the core concept of our approach. We generalize the string representation of motifs as follows.

**DEFINITION 1.** For some pre-defined coverage ratio  $r \leq 1$ , a spaced motif (or simply a motif) is a length- $L$  string formed

by characters of  $\{A, C, G, T, n\}$  with at least  $\lfloor r \times L \rfloor$  characters in  $\{A, C, G, T\}$ . Each maximal substring of consecutive 'n' represents a spacer and each maximal substring of other characters represents a segment.

Figure 1 shows an example of a spaced motif  $M$  which is of length 20 and has three segments separated by two spacers. Note that the segments, as well as the spacers, can be of different lengths. The number of segments is also not fixed. Let  $Z[i..j]$  be the substring of  $Z$  starting at position  $i$  and ending at position  $j$ . Any length- $\ell$  substring  $M[i..i + \ell - 1]$  within any segment of  $M$  is called its *submotif*. Below, using submotifs, we define an instance of a spaced motif (see Fig 1 for an example).

**DEFINITION 2.** Consider a length- $L$  spaced motif  $M$  and any length- $L$  string  $I$  formed by characters of  $\{A, C, G, T\}$ .  $I$  is called an instance of  $M$  if, for every submotif  $M[i..i + \ell - 1]$  and  $I[i..i + \ell - 1]$  have at most  $d$  mutations, for some pre-defined constant  $d$ .

Now, we define the *spaced motif-finding problem*. Let  $S = \{s_1, s_2, \dots, s_l\}$  be a given set of DNA sequences. Our task is to identify spaced motifs with at least  $q$  instances in  $S$ , for some predefined constant  $q$  (we call this the *minimum support*). Figure 2 shows an example.

By formulating the motif-finding problem in this way, we have the following advantages:

- (1) The lengths of the segments in the motif need not be known *even if* we pre-fix the length of the submotif. This follows because union of an overlapping set of submotifs can represent an arbitrary length segment. This property implies that motifs with segments of arbitrary lengths could be found. Note that this does not depend on whether the motif has spacers or not.
- (2) The spaced motif uses multiple segments to model the functional parts, which are more conserved, and the spacers to model the non-functional parts. However, monad motif (or dyads) only has one segment (or two

```
M=CAGTTTCnACGTCnnGACGT
I1=TAGTTTAtATGTCcgGACAT
I2=CACTTTAAtATGTCcgCACGT
```

**Fig. 1.** Consider  $L=20$ ,  $\ell=5$  and  $d=1$ .  $M$  is an example of length-20 spaced motif with three segments separated by two spacers. Then,  $I_1$  is an instance of  $M$  since all length-5 submotifs in the three segments of  $M$  have less than 1 mismatches when comparing with  $I_1$ . On the other hand,  $I_2$  is not an instance of  $M$  since  $M[2..6]$  and  $I_2[2..6]$  have two mismatches.

```
TTGATACCGAAGATACCGATTAGAAATCACTCA
ACTACAGAAAAGCAGTAGTAAAAACGTACAGTC
GAAGACCGTCATGAGAAATCGCATAACAGGCA
TTCACCCGATAAAAAATAAGGCTGTCTGGACTAA
TCGGAAACAATTACGAAGAAAAGCAGTAGAAAAA
```

**Fig. 2.** Consider  $L=20$ ,  $r=0.5$ ,  $\ell=5$ ,  $d=1$  and  $q=4$ . GAAGAnnnnnTAGAAAnn is a spaced motif of the above five sequences. All its instances are underlined.

segments) for modeling both conserved and non-conserved regions. Hence, spaced motif can fit the conserved regions better. In other words, it yields higher specificity. We confirm this in our experiments on several datasets including the Motif Assessment Benchmark and some real biological datasets.

- (3) It provides a natural extension for finding motifs with multiple spacers, in which neither the spacer length nor the number of spacers (and segments) is known.

However, there could be too many submotifs (many of them are spurious) and the challenge is in how effectively the submotif-compositing can be done to return ‘good’ motifs. To tackle this situation, we formulate this task as a constrained frequent submotif mining problem and propose a new algorithm for solving it.

### 3 OUR SOLUTION

Let  $S = \{s_1, s_2, \dots, s_t\}$  be the given set of  $t$  sequences. Our solution for finding spaced motifs is called *SPACE*. It consists of three main steps. Step 1 finds motif candidates, which is defined below. Step 2 refines the motif candidates into spaced motifs. Lastly, Step 3 computes the significance of the spaced motifs based on our scoring function and reports the ranked list of motifs.

We do not assume any knowledge about the number and the locations of the spacers in the motif. To identify a possible candidate for the motif, we look at each length- $L$  substring  $u$  in  $S$ , based on the definition of a spaced motif, we define an occurrence of  $u$  as follows. Let  $hd(x, y)$  be the Hamming distance of two equal-length strings  $x$  and  $y$ .

**DEFINITION 3.** Let  $u$  be a length- $L$  substring in  $S$ . Consider another substring  $w$  of the same length in  $S$ . For some pre-defined constants  $d$  and  $r \in [0, 1]$ , for every  $i$ , the substring  $w[i..i + \ell - 1]$  is called a submotif occurrence of  $u[i..i + \ell - 1]$  if  $hd(u[i..i + \ell - 1], w[i..i + \ell - 1]) \leq d$ . The number of characters spanned by all submotif occurrences is called the coverage of  $w$  on  $u$ . The substring  $w$  is called an occurrence of  $u$  if the coverage of  $w$  on  $u$  is at least  $\lfloor r \times L \rfloor$ .

The length- $L$  substring  $u$  is called a *motif candidate* if there exist at least  $q$  occurrences of  $u$  in  $S$ . Figure 3 shows an example of a motif candidate.

Step 1 tries to find all motif candidates. A straightforward implementation is given in Section 3.1. Note that a spaced motif is highly correlated with a motif candidate. For a motif candidate that is a real spaced motif, the locations of submotif

```

TTGATACCGAAGATACCGATTAGAAATCACTCA
ACTACAGAAAAGCAGTAGTAAAAACGTACAGTC
GAAGACCGTCATGAGAAATCGCATAACCGAGCA
TTCACCCGATAAAAAATAAGGCTGTCTGGACTAA
TCGGAACAATTACGAAGAAAAGCAGTAGAAAAA

```

**Fig. 3.** Consider  $L=20$ ,  $r=0.5$ ,  $\ell=5$ ,  $d=1$  and  $q=4$ . For the same set  $S$  of 5 sequences in Figure 2, GAAGATACCGATTAGAAATC has five occurrences. All its occurrences are underlined. Since the number of occurrences is at least  $q$ , it is a motif candidate.

occurrences in each occurrence of the motif candidate define the locations of the segments for the candidate. A different occurrence may define a different set of segments for the same candidate. By finding the set of common segments defined by the occurrences, we can generate a spaced motif. The refinement process is done in Step 2 based on frequent itemset mining, which is detailed in Section 3.2. Step 3 and our scoring function is discussed in Section 3.3. The naive implementation shown in Section 3.1 is a bit slow, Section 3.4 shows how to speed up the process.

#### 3.1 Generation of motif candidates

To find all motif candidates and their occurrences, a straightforward implementation is as follows. Fix a constant  $L$  for the motif length, for each sequence  $S_i$ , for each substring  $u$  of length  $L$  in  $S_i$ , check the coverage of all other substrings in  $S$  of length  $L$  on  $u$ . If there are  $q$  occurrences of  $u$ , report  $u$  and all its occurrences. This naive procedure runs in  $O(Ln^2)$  time where  $n$  is the length of a sequence. The actual running time is about 2min for a dataset of 5K bp with 10 sequences on a 3.6GHz Xeon Linux workstation with 4 processors and 8GB RAM.

At the end of this step we have a set of motif candidates, each associated with a set of occurrences. Recall that some of these occurrences may be noise or some of the submotif occurrences in them may be spurious. Our next step is to eliminate these noise to identify the spaced motifs.

#### 3.2 Refining motif candidate into spaced motif

Given a motif candidate  $u$  and its occurrences  $w_1, w_2, \dots, w_c$ , this section discusses the way to refine  $u$  into a spaced motif. Our idea is to transform the problem into frequent itemset mining (Han and Kamber, 2000).

Before describing the transformation, recall that, by Definition 3,  $u$  and  $w_i$  share a set of submotif occurrences.

**DEFINITION 4.** Suppose that  $w$  is an occurrence of  $u$ . Then,  $\{j \mid 1 \leq j \leq L - \ell + 1, hd(w[j..j + \ell - 1], u[j..j + \ell - 1]) \leq d\}$  is called the itemset of  $w$  with respect to  $u$ .

Figure 4 demonstrates the itemset concept. For an itemset  $J$ , we can construct a spaced motif  $M_{u,J}$  of length  $L$  such that  $M_{u,J}[i] = u[i]$  if  $0 \leq i - j < \ell$  for some  $j \in J$ ; otherwise,

```

GAAGATACCGATTAGAAATC :
    {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16}
GAAAAGCAGTAGTAAAAACG :
    {1, 13, 14}
GAAGACCGTCATGAGAAATC :
    {1, 11, 12, 13, 14, 15, 16}
CCCGATAAAAAATAAGGCTGT :
    {3, 4, 12}
GAAGAAAAGCAGTAGAAAAA :
    {1, 2, 13, 14}

```

**Fig. 4.** Consider  $L=20$ ,  $r=0.5$ ,  $\ell=5$ ,  $d=1$  and  $q=4$ . With respect to the sequence set  $S$  in Figure 3, this figure shows the five occurrences of GAAGATACCGATTAGAAATC and their corresponding itemsets. Note that  $\{1, 13, 14\}$  is the frequent itemset which appears four times. Hence, GAAGAnnnnnnTAGAAAnn is a spaced motif of the set  $S$ .

$M_{u,J}[i] = n$ . An itemset is called a frequent pattern if it has at least  $q$  occurrences. The following lemma states the relationship between frequent itemset and spaced motif. Note that there is an assumption behind this transformation. While we allow different gaps to have different lengths in the same motif, for a gap in the motif, the length of this gap is the same in all instances.

**LEMMA 1.** *Let  $J$  be a frequent pattern of  $u$  with at least  $q$  support. If  $M_{u,J}$  has coverage at least  $\lfloor r \times L \rfloor$ ,  $M_{u,J}$  is a spaced motif.*

**PROOF.** Since  $J$  is a frequent pattern with at least  $q$  support,  $M_{u,J}$  has at least  $q$  instances. Also,  $M_{u,J}$  has coverage at least  $\lfloor r \times L \rfloor$ , so  $M_{u,J}$  is a spaced motif. ■

Hence, given a motif candidate  $u$  and its occurrences, we can refine  $u$  as a spaced motif as follows.

- (1) Generate the itemsets for all occurrences of  $u$ .
- (2) Find the frequent itemsets  $F$  which appear at least  $q$  times.
- (3) Report the spaced motif corresponding to  $F$  with sufficient coverage.

Algorithm 1 shows the complete scheme of the algorithm.

---

#### Algorithm 1 SPACE

---

**Input:**  $l_s, d, e, q, r, S$

**Output:** Ranked motifs

- 1: from  $S$  generate the set of motif candidates ( $D$ ), each associated with a set of occurrences
  - 2: **for** each motif candidate  $u$  in  $D$  **do**
  - 3: Let  $W$  be the set of occurrences of  $u$
  - 4: Find all frequent patterns that appear in  $W$
  - 5: **for** each frequent pattern **do**
  - 6: construct the corresponding spaced motif  $M$
  - 7: If  $M$  has enough coverage, keep and score  $M$
  - 8: **end for**
  - 9: **end for**
  - 10: return ranked spaced motifs
- 

### 3.3 Significance testing and scoring

We adapt the motif-scoring technique introduced in Weeder (Pavesi *et al.*, 2001) to compute the significance of spaced motifs. Intuitively, a motif is significant if (1) the total number of its occurrences in all input sequences is a lot more than expected with respect to the background and (2) the pattern is either very conserved or occurs in quite a number of the input sequences. So, Weeder's scoring mechanism computes two values to capture these two properties.

Let  $M$  be the motif,  $E(M, e)$  be the expected frequency of  $M$  with at most  $e$  mutations based on a set of background sequences (we will show how to compute  $E$  later in this section). Then,  $E(M, e) \cdot \sum \text{len}(s_i)$ , where  $\text{len}(s_i)$  denotes the length of  $i$ -th sequence  $s_i$ , represents the expected frequency of  $M$  with at most  $e$  mutations in all input sequences. To capture property (1), we count the total number of observed occurrences of  $M$

(with at most  $e$  mutations),  $Occ_s(M, e)$ , in all input sequences and compute the *occurrence score*,  $\beta(M)$  as follows.

$$\beta(M) = \log \frac{Occ_s(M, e)}{E(M, e) \cdot \sum \text{len}(s_i)} \quad (1)$$

To capture property (2), for a sequence  $s'_i$  with an occurrence of  $M$ , we consider the most conserved pattern of  $M$  and let  $e_i$  be the number of mutations of this best pattern. The value of  $E(M, e_i) \cdot \text{len}(s'_i)$  represents the expected frequency of the occurrences of this motif in  $s'_i$ . This value is smaller if the motif is more conserved. Then, we compute the *sequence-specific score*,  $\sigma(M)$  as follows. If the pattern is very conserved and/or occurs in many sequences,  $\sigma(M)$  is large.

$$\sigma(M) = \sum_i \log \frac{1}{E(M, e_i) \cdot \text{len}(s'_i)} \quad (2)$$

Finally the score of each motif, *Motif Score*( $M$ ), is  $\sigma(M) + \beta(M)$ .

The value of  $E(M, e)$  is computed by summing the expected frequency  $E(M')$  of  $M'$  in the background sequences for all  $M'$  with at most  $e$  mutations from  $M$ . When  $M'$  contains no spacer and is of length shorter than or equal to 8, the expected frequency value  $E(M')$  is pre-computed from background sequences obtained from Regulatory Sequence Analysis Tool (RSAT) database site<sup>1</sup> (van Helden, 2003). These background sequences of the organisms are taken from 1000 bp upstream regions of all their annotated genes.

When  $M'$  contains spacers and is of length shorter than or equal to 8,  $E(M')$  equals the sum of  $E(M'')$  among all possible  $M''$  with the spacers  $n$ 's replacing by  $\{A, C, G, T\}$ .

When  $M'$  is of length longer than 8 and with or without spacers, we are unable to precompute the frequency values since it is long. Instead, the expected frequency of  $M'$  is modelled using seventh order Markov chain. Suppose  $M' = p_1 p_2 \dots p_k$  with  $k$  greater than 8.  $E(M')$  can be computed as follows:

$$E(M') = E(p_1 p_2 \dots p_8) \prod_{i=9}^k P(p_i | p_{i-7} \dots p_{i-1})$$

The conditional probability  $P(p_i | p_{i-7} \dots p_{i-1})$  of having nucleotide  $p_i$  preceded by nucleotides  $p_{i-7} \dots p_{i-1}$ , is computed by using the expected frequency of 8-mers:

$$P(p_i | p_{i-7}, \dots, p_{i-1}) = \frac{E(p_{i-7} \dots p_i)}{E(p_{i-7} \dots p_{i-1} n)}$$

### 3.4 Efficient generation of motif candidates

This section shows how to speed up Step 1, the motif candidate generation step. The observations that lead to the speed up are as follow. Recall that  $\ell$  is the length of a submotif.

---

<sup>1</sup><http://rsat.ulb.ac.be/rsat>

LEMMA 2. Let the coverage of  $S_a[b..b+L-1]$  on  $S_i[j..j+L-1]$  be  $C$ . Then, the coverage  $C'$  of  $S_a[b+1..b+L]$  on  $S_i[j+1..j+L]$  can be computed as follows.

$$C' = \begin{cases} C+1 & \text{if } \alpha = 0 \text{ and } \beta = 1 \\ C & \text{if } \alpha = \beta = 0 \text{ or } \alpha = \beta = 1 \\ C-1 & \text{if } \alpha = 1 \text{ and } \beta = 0 \end{cases}$$

where  $\alpha = 1$  if the prefix  $S_i[j..j+\ell-1]$  of  $S_i[j..j+L-1]$  is a submotif occurrence, that is,  $hd(S_i[j..j+\ell-1], S_a[b..b+\ell-1]) \leq d$ , otherwise  $\alpha = 0$ . Similarly,  $\beta = 1$  if the suffix  $S_i[j+L-\ell+1..j+L]$  of  $S_i[j+1..j+L]$  is a submotif occurrence, that is,  $hd(S_i[j+L-\ell+1..j+L], S_a[b+L-\ell+1..b+L]) \leq d$ , otherwise  $\beta = 0$ .

PROOF. Note that when considering all length- $\ell$  substrings of  $S_i[j+1..j+L]$  and  $S_i[j..j+L-1]$ , the only substrings that they are different are  $S_i[j..j+\ell-1]$  which is in  $S_i[j..j+L-1]$ , but not in  $S_i[j+1..j+L]$ , and  $S_i[j+L-\ell+1..j+L]$  which is in  $S_i[j+1..j+L]$ , but not in  $S_i[j..j+L-1]$ .

If  $\alpha = 1$ , it means that  $S_i[j..j+\ell-1]$  is a submotif of  $S_i[j..j+L-1]$  with respect to  $S_a[b..b+L-1]$ . This submotif will not be in  $S_i[j+1..j+L]$ . If  $\beta = 1$ , then  $S_i[j+L-\ell+1..j+L]$  is a submotif of  $S_i[j+1..j+L]$  with respect to  $S_a[b+1..b+L]$  which is not in  $S_i[j..j+L-1]$ .

So, the result follows. ■

Based on Lemma 1, once we have calculated the coverage of  $S_a[b..b+L-1]$  on  $S_i[j..j+L-1]$ , to calculate the coverage of  $S_a[b+1..b+L]$  on  $S_i[j+1..j+L]$ , it only takes  $O(1)$  time. To calculate the coverage of all substrings on one sequence against all potential motif candidates in another sequence, the time complexity can then be reduced to  $O(n^2)$ .

Since we are only interested in the substrings that can have a coverage at least  $\lfloor r \times L \rfloor$ , we can further prune the computation according to the following lemma.

LEMMA 3. Let the coverage of  $S_a[b..b+L-1]$  on  $S_i[j..j+L-1]$  be  $C$ . Let  $y$  be the length of the longest suffix of  $S_i[j..j+L-1]$  that is not covered by a submotif occurrence. The coverage  $C'$  of  $S_a[b+p..b+p+L-1]$  on  $S_i[j+p..j+p+L-1]$  is upper bounded by  $C + \min\{y, \ell - 1\} + p$  for any  $p > 0$ .

PROOF. We try to upper bound the value of  $C'$  as follows. Comparing  $S_i[j+p..j+p+L-1]$  with  $S_i[j..j+L-1]$ , there are  $p$  new characters. Assuming that all these characters are covered by submotifs, the coverage can be increased at most by  $p$ . For the suffix of  $S_i[j..j+L-1]$  that is not covered by any submotif occurrence. If  $y < \ell - 1$ , then when considering  $S_i[j+p..j+p+L-1]$ , these  $y$  characters may all be covered by a submotif, so the coverage can be increased by at most  $y$ . On the other hand, if  $y \geq \ell - 1$ , then at most the last  $\ell - 1$  characters, which can form a submotif with one new character, can be covered by a submotif occurrence when considering  $S_i[j+p..j+p+L-1]$ , so the coverage can be increased by at most  $\ell - 1$ . So, the result follows. ■

By Lemma 2, after computing the coverage of  $S_a[b..b+L-1]$  on  $S_i[j..j+L-1]$ , based on the upper bound calculation, we can skip the computation of coverage for some substrings and jump to the substrings  $S_a[b+p..b+p+L-1]$

and  $S_i[j+p..j+p+L-1]$  with the smallest  $p$  such that  $C + \min\{y, \ell - 1\} + p \geq \lfloor r \times L \rfloor$ . From our experiments, we found that the running time for generating the motif candidates and their occurrences have been reduced from 2 min to  $< 1$  s on the same dataset of 5K bp with 10 sequences on a 3.6GHz Xeon Linux workstation with 4 processors and 8GB RAM. So, it is feasible for large datasets.

## 4 EXPERIMENTAL RESULTS

We perform experiments on four classes of datasets namely, (1) nine biological datasets that are known to contain spacers, (2) four synthetic test cases consisting of different variations of spaced motifs, (3) The datasets from four different species, proposed by (Tompa *et al.*, 2005) for the assessment of motif discovery algorithms and (4) 10 real biological datasets consisting monad motifs. The assessment results are reported below.

For performance evaluation, we use the same four measures proposed in (Tompa *et al.*, 2005) namely, *sensitivity* ( $nSn$ ), *positive predictive value* ( $nPPV$ ), *performance coefficient* ( $nPC$ ), and *correlation coefficient* ( $nCC$ ). Index  $n$  is used to denote that the assessment is done at the nucleotide level instead of site level (please refer to Appendix B for the definitions of these measures).<sup>2</sup> All experiments have been performed on a 3.6GHz Xeon Linux workstation with 4 processors and 8GB RAM.

For each dataset, we run SPACE using 12 parameter settings: motif length  $L = 8, 15, 20$ ; submotif length  $\ell = 5$ ; maximum number of mismatches allowed in each submotif instance  $d = 1$ ; the minimum support  $q = t$  or  $0.5t$ , where  $t$  is the number of input sequences; the coverage ratio  $r$  is set to be 0.5 or 0.8. Similar to what Weeder does, we collect the top 10 motifs from each run. For each motif, we count the number of related redundant motifs in the collection. Based on the observation that a real motif should have more related redundant motifs, we rank the motifs in decreasing order of the total number of related redundant motifs in the collection. Please refer to Appendix A for a more detailed description of this step.

### 4.1 Results on datasets with spaced motifs

We first evaluate the performance of SPACE for spaced motifs, that is, motifs with at least one spacer.

**4.1.1 Real biological datasets** In the literature, we found nine transcription factor binding sites whose motifs have gaps. They are: GAL4P (Johnston and Carlson, 1992), ARCA-P (Liu and DeWulf, 2004), MCM1 (Kato *et al.*, 2004) or ECB (Tavazoie *et al.*, 1999) and six transcriptional regulators of C6 Zinc cluster family (Schjerling and Holmberg, 1996).

<sup>2</sup>There is no consensus on what measures are the most appropriate to evaluate all different motif finders. The selected measures focus on the accuracy of predicting the locations of actual binding sites. Note that the consensus of the motifs reported may be the same for different algorithms, but the predicted binding sites may be different, thus yielding a difference in the performance measures.

**Table 1.** Comparison of SPACE, MITRA and BioProspector on spaced motifs in real biological datasets (the first motif among the top 20 that gives a better nSn and nPPV than motif of Rank 1 is used for comparison)

TF		Motif	RANK	nSn	nPPV	nCC	nPC
GAL4P (Johnston and Carlson, 1992)	Actual	CGGRnnRCYnYnChCCG					
	SPACE	CGGAnGACTnnnnTCCG	1	0.80	0.55	0.48	0.65
	MITRA	AGCGGnnGACTC	1	0.68	0.40	0.34	0.51
ARCA-P (Liu and DeWulf, 2004)	BioProspector	TCCGnnnnnnnnnnCCGT	1	0.63	0.26	0.23	0.39
	Actual	GTTAAnnnnnnGTTAA					
	SPACE	GTTAnnnnnATGTTA	1	0.80	0.59	0.52	0.68
ECB (Kato <i>et al.</i> , 2004) (Tavazoie <i>et al.</i> , 1999)	MITRA	GTTAACT	15	0.60	0.32	0.26	0.42
	BioProspector	GTTATnnnnnnnTAAA	4	0.66	0.25	0.22	0.38
	Actual	TACCnAATtnGGTAA					
CAT8 (van Helden <i>et al.</i> , 2000)	SPACE	TTACnnAATtnGGAA	1	0.70	0.58	0.46	0.61
	MITRA	CCAAntTtnGnGAA	2	0.61	0.48	0.36	0.51
	BioProspector	TCCTAnnnnGGAAA	2	0.72	0.33	0.30	0.47
HAP1 (Schjerling and Holmberg, 1996) (Svetlov and Cooper, 1995)	Actual	CGGnnnnnnGGA					
	SPACE	CGGAnnnnnGGAAAT	1	0.74	0.52	0.44	0.62
	MITRA	CCGTnGTTCCGGA	5	0.57	0.31	0.25	0.40
LEU3 (Svetlov and Cooper, 1995)	BioProspector	CGGAnnnnCGGG	1	0.64	0.40	0.33	0.50
	Actual	CGGnnTAnCGGnnnTA					
	SPACE	CCGGnVTTtnCGGH	2	0.67	0.67	0.50	0.66
LYS (Becker <i>et al.</i> , 1998)	MITRA	CGGATnTnCCGG	1	0.67	0.18	0.17	0.33
	BioProspector	GCGGnnnnnnCGGA	5	0.85	0.15	0.14	0.34
	Actual	RCCGGnnCCGGY					
PPR (Schjerling and Holmberg, 1996)	SPACE	CCGGnnCCGGCT	1	0.85	0.28	0.27	0.48
	MITRA	CGGnACCGAnGC	2	0.46	0.13	0.11	0.22
	BioProspector	CCGGnnCCGG	1	0.71	0.19	0.17	0.35
PUT3 (Schjerling and Holmberg, 1996)	Actual	WWWTCRRnYGGAWWW					
	SPACE	AATTCGnnGGAA	4	0.62	0.59	0.43	0.60
	MITRA	TCCACnGGAA	4	0.75	0.33	0.30	0.48
AVERAGE	BioProspector	ATTTcnAGCGG	3	0.56	0.27	0.22	0.37
	Actual	WYCGGnnWYKCCGAW					
	SPACE	TCCGnnnnnnGCCGAAG	1	0.88	0.75	0.68	0.81
AVERAGE	MITRA	CGGnTTCtnCG	9	0.67	0.36	0.30	0.47
	BioProspector	TCGGCnnTCTCCGA	1	0.78	0.52	0.45	0.63
	Actual	YCGGnAnCGGnAnnnCCGA					
AVERAGE	SPACE	TCGGGAnnnnnnnTCCG	1	0.89	0.76	0.69	0.81
	MITRA	TCGGnAnCCGAA	2	0.75	0.62	0.52	0.66
	BioProspector	TCGGAnnnnnnnnnCCGGA	2	0.64	0.64	0.47	0.62
AVERAGE	SPACE			0.77	0.59	0.50	0.66
	MITRA			0.64	0.35	0.29	0.44
	BioProspector			0.69	0.33	0.28	0.45

Comparison is done with MITRA<sup>3</sup> (Eskin and Pevzner, 2002) and BioProspector,<sup>4</sup> both of which can handle motifs with spacers. We let MITRA search for motifs up to 12bp (the maximum possible) and we require it to find the motif on the given strands only. For BioProspector we allow the algorithm to search for motifs with block size ranging from 4 to 10 and gap size ranging from 0 to 12. In the comparison, instead of picking the first motif among the top 20 that can give a better nSn and nPPV than the motif of rank 1. Table 1 summarizes the comparison results. From the table, we see that the selected motifs of SPACE are usually of higher rank than

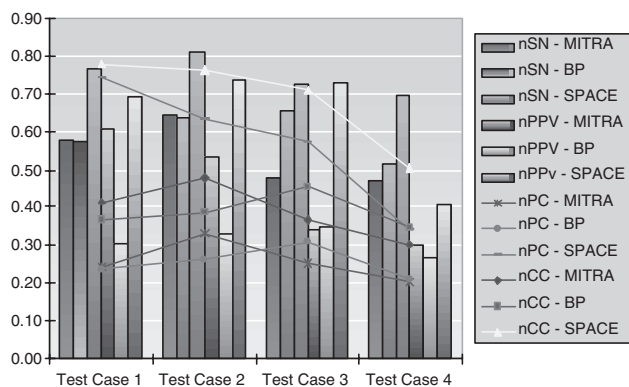
the other two and the averaged performance is better across all measures.

**4.1.2 Synthetic datasets** We consider four synthetic test cases for spaced motifs using randomly created sequences with the base pairs uniformly distributed. For each case, we create three datasets, each containing 10 sequences of length 300bp. We run SPACE and report the *averaged* performance. For each dataset the motifs are implanted in five of them at random positions. And the motifs are as follow:

- (1) A 7bp length motif with no spacer and 1 mismatch.
- (2) A 15bp length motif containing two segments of length 5 and 7 with a spacer of length 3, with 1 mismatch for each segment.

<sup>3</sup><http://fluff.cs.columbia.edu:8080/domain/mitra.html>

<sup>4</sup><http://ai.stanford.edu/~xslu/BioProspector/>



**Fig. 5.** Comparison of MITRA, BioProspector (denoted BP) and SPACE averaged performance on four motif-finding problems. Colour version of this figure is available as Supplementary material online.

- (3) A 21bp length motif containing two segments of length 5 with spacers of length 3, with 1 mismatch for each segment.
- (4) A 15bp length motif containing two segments of length 4 and 5 bp with a spacer of length 6, with 1 mismatch for each segment.

Figure 5 shows the averaged performance of SPACE, MITRA and BioProspector on the synthetic datasets with Table 3 giving the detailed statistics on one particular dataset of each test case. It also shows that SPACE performs better than the other tools over all four measures. The result is consistent with the one for real biological datasets. More information about the comparison results can be obtained in Appendix L of the Supplementary Material.

## 4.2 Results on datasets with monad motifs

We are also interested in the performance of SPACE for motifs without spacers. We have performed two sets of experiments, one on Tompa's benchmark datasets and the other on 10 real biological datasets.

**4.2.1 Tompa's benchmark data** Tompa's benchmark dataset has been constructed based on real transcription factor binding sites drawn from four different organisms (Tompa *et al.*, 2005). It consists of 56 datasets in total. The number of sequences ranges from 1–35 and the sequence lengths are up to 3000 bp. In this assessment, following Tompa's approach, the motif ranked number 1 by the algorithm is used for comparison. The detailed experimental results can be found in Appendix C of Supplementary Material.

The performance of SPACE averaged over all datasets is shown in Figure 6. SPACE performs better than other tools based on the comparison measures.<sup>5</sup> As an example, we show the binding sites (see Fig. 7) identified (in green) by our algorithm and Weeder on the dataset *hm17g* together with

the actual binding sites (in blue). Weeder is reported to perform the best among other tools in this dataset (Tompa *et al.*, 2005). Similar to Weeder, SPACE is able to identify almost all actual binding sites.

We also analysed the performance of SPACE across the four organisms. Figure 8 shows the average performance of SPACE for each organism, compared with the best algorithm among the other tools for the respective organism. The figure shows that the performance of SPACE is similar to that of the best performing algorithm for each organism. On the other hand, the averaged performance of SPACE for all four organisms is better than other tools (Fig. 6), indicating that SPACE is more robust and organism independent.

**4.2.2 Real biological datasets** We also performed experiments on 10 real biological datasets whose binding sites are known to be monads from literature. Comparison is done with Weeder (Pavesi *et al.*, 2001) and MEME (Bailey and Elkan, 1995), both of which are well-known monad motif-finding algorithms. We set MEME to use two-component mixture mode and find motifs of length ranging from 8 to 20 bp. And for Weeder we use the *large* mode. Table 3 summarizes the comparison results. From the table, for sensitivity, MEME shows a better performance than Weeder while SPACE is better than MEME. The reason for SPACE to have a higher sensitivity is due to the submotif modelling. Since the measure nSn focuses on predicting the binding sites, if there is a region in the motif that is not strongly conserved over all binding sites, the use of submotifs may still be able to identify most of these binding sites based on the regions that are strongly conserved, thus predicting more true binding sites. However, it does not mean that missing these binding sites, the software will predict a wrong motif pattern.

Unlike the case for spaced motifs, SPACE is not a clear winner for all the measures except sensitivity. But, the experiments indicate that for monads, SPACE is as good as other tools. This shows that SPACE can be used as a standard tool for finding monads as well as spaced motifs.

The real biological datasets of the spaced and monad motifs are constructed from –1000 to –1 upstream region of all the genes co-regulated by respective transcription factor, truncating the region if it overlaps with an upstream open reading frame (ORF). They are obtained from RSAT (van Helden *et al.*, 2000) and ABS (Blanco *et al.*, 2006) database respectively. More details on the results can be found in Appendices E and F.

## 5 CONCLUSIONS

In this article, we have proposed a new approach for finding spaced motifs based on the notion called submotifs. We developed a novel motif-finding algorithm SPACE that detects the target motif by first finding submotifs and then strategically compositing them using an efficient frequent submotif pattern-mining approach. In finding motif with generic spacers, this framework provides the following novelties: the spacers could appear in more than two parts of the motif and their lengths need not be fixed. In experiments on real biological datasets, synthetic datasets and Tompa's motif assessment benchmarks, we observed that our algorithm performs better than existing

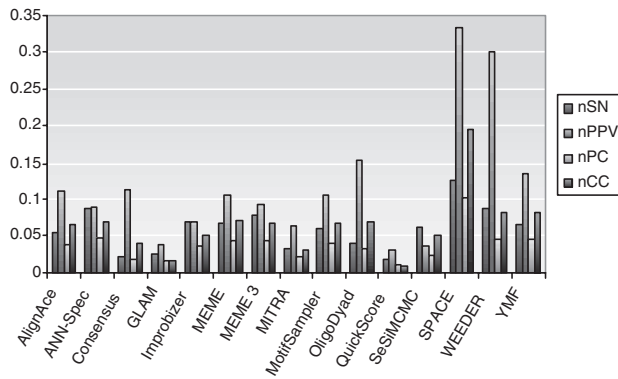
<sup>5</sup>In our comparison, we did not include the new motif finder, MotifSeeker (Peng *et al.*, 2006). The experiments in their paper are based on a different subset of Tompa's datasets, so a direct comparison is not appropriate.

**Table 2.** Performance of SPACE, MITRA and BioProspector (denoted BP) on four types of synthetic data (one dataset each)

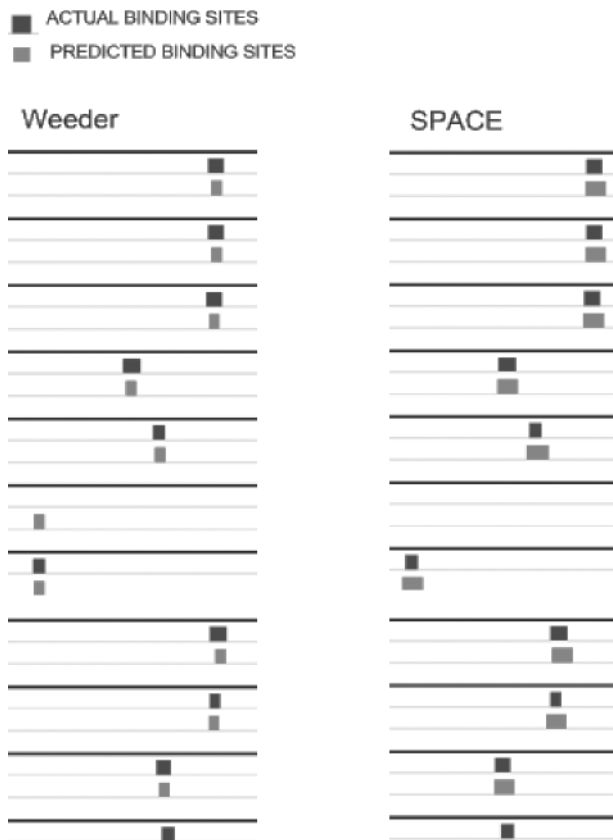
Problem	Motif	RANK	nSn	nPPV	nPC	nCC
1. Actual	TGGGTAC					
SPACE	GGGTACC	3	0.83	0.72	0.75	0.82
MITRA	GGTACCCn	5	0.57	0.64	0.33	0.57
BP	GGGTACC	1	0.62	0.32	0.27	0.44
2. Actual	CCTGTnnnAGTTGTC					
SPACE	CCTGTnnTAGTTG	1	0.81	0.76	0.65	0.78
MITRA	CnTGTACTIONGTT	2	0.67	0.68	0.29	0.44
BP	CCTGTnnnACTTGTT	2	0.67	0.31	0.27	0.44
3. Actual	ATCGTnnnnTGACCnnnnCTTTC					
SPACE	TCGTnnnnTGACCnnnnTTTC	1	0.76	0.66	0.55	0.69
MITRA	ATCCnTnTGAC	1	0.49	0.38	0.27	0.39
BP	ATCGTnnnnnnnnnnCTTTC	1	0.71	0.38	0.33	0.50
4. Actual	CGGCnnnnnnTCTAA					
SPACE	TTCGGYnnnnTGTC	1	0.71	0.39	0.33	0.50
MITRA	CGGCnAAGnGTC	3	0.50	0.24	0.13	0.20
BP	CGTAnnnnnTCTAA	1	0.33	0.18	0.13	0.22

**Table 3.** Comparison of SPACE, MITRA and Weeder on monads in real biological datasets (the first motif among the top 20 that gives a better nSn and nPPV than motif of Rank 1 is used for comparison)

TF		Motif	RANK	nSn	nPPV	nCC	nPC
AP2A (Lenhard <i>et al.</i> , 2003) (Dermitzakis and Clark, 2002)	Actual	GCCGGGGKSG					
	SPACE	CCAGGGAG	1	0.75	0.50	0.43	0.60
	MEME	GCCCCCCC	5	0.38	0.56	0.29	0.45
	Weeder	CCCCACCC	3	0.38	0.33	0.21	0.34
CAAT (Wasserman and Fickett, 1998)	Actual	AAGCCAATTAGGCC					
	SPACE	GAAGCCAATTAG	1	0.51	0.60	0.38	0.54
	MEME	CGAAGCAA	2	0.08	0.67	0.08	0.20
	Weeder	GCCAAT	1	0.63	0.78	0.54	0.70
CJUN (Han <i>et al.</i> , 1992)	Actual	ATTATTCACHTCATC					
	SPACE	CATTWCCTCA	1	0.64	0.73	0.52	0.68
	MEME	CATTACCTCA	2	0.62	0.81	0.55	0.71
	Weeder	CATTACCTCA	3	0.62	0.81	0.55	0.71
CMYC (Hermeking <i>et al.</i> , 2000)	Actual	CACGTG					
	SPACE	CACGTGCC	1	1.00	0.60	0.60	0.77
	MEME	GGTCACGTGGGATAGCAACA	1	1.00	0.27	0.27	0.71
	Weeder	TCACGT	1	0.71	0.71	0.56	0.71
E2F (Yagi <i>et al.</i> , 1995)	Actual	TTTCGCGC					
	SPACE	TTTCGCGCC	1	0.91	0.81	0.80	0.88
	MEME	TTGTGCGGCC	4	1.00	0.82	0.82	0.90
	Weeder	TTTCGCGC	2	0.64	0.91	0.60	0.75
ETS1 (Dermitzakis and Clark, 2002)	Actual	KAGGAAGT					
	SPACE	AGGAAGTA	1	0.76	0.65	0.54	0.70
	MEME	GGTATTCA	3	0.62	0.56	0.42	0.58
	Weeder	AAGTAG	1	0.40	0.48	0.28	0.43
GC (Dermitzakis and Clark, 2002)	Actual	GGGCGGCC					
	SPACE	GCCCCTGCC	1	0.56	0.60	0.41	0.57
	MEME	AAGGCTGCGTGGAC	1	0.57	0.27	0.22	0.38
	Weeder	ACCCAC	5	0.62	0.71	0.50	0.66
MTF1 (Blanchette and Tompa, 2002)	Actual	GGGTGCACTCG					
	SPACE	GCACACTGGC	3	0.71	0.36	0.31	0.50
	MEME	TGCAAAACCCFTTGCGCC	6	0.65	0.29	0.25	0.42
	Weeder	CTCGTA	9	0.38	0.43	0.25	0.39
MYB (Lenhard <i>et al.</i> , 2003)	Actual	GAACGTTA					
	SPACE	CGTTACG	1	0.71	0.50	0.42	0.59
	MEME	ACGTTACGAA	9	1.00	0.55	0.55	0.73
	Weeder	GTTACG	1	0.57	0.57	0.57	0.40
MYF (Lenhard <i>et al.</i> , 2003)	Actual	GGGCCAGTTGTCCC					
	SPACE	GGGGCCAGG	2	0.54	0.71	0.44	0.61
	MEME	GGCAAGCAG	5	0.39	1.00	0.39	0.62
	Weeder	CTGGGTCGAC	1	0.47	0.64	0.37	0.53
AVERAGE	SPACE			0.71	0.61	0.49	0.64
	MEME			0.63	0.58	0.38	0.46
	Weeder			0.54	0.64	0.43	0.58



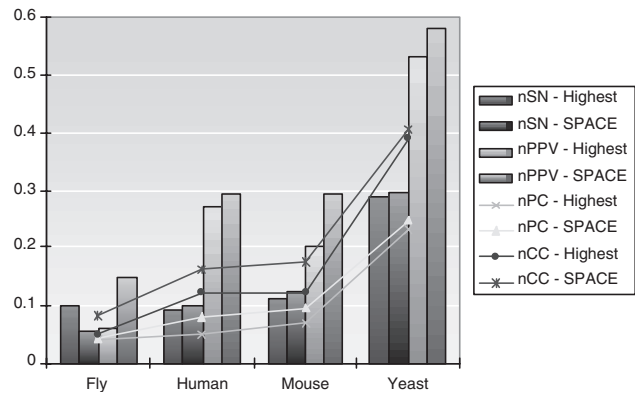
**Fig. 6.** Comparison between SPACE with 13 other motif discovery tools. Colour version of this figure is available as Supplementary material online.



**Fig. 7.** Binding sites without gaps reported by SPACE in *hm17g* (human), with  $nSn = 0.90$ ,  $nPPV = 0.72$ ,  $nPC = 0.67$  and  $nCC = 0.80$ . Weeder with  $nSn = 0.61$ ,  $nPPV = 0.89$ ,  $nPC = 0.57$  and  $nCC = 0.73$ . Colour version of this figure is available as Supplementary material online.

tools for spaced motifs with improvements in both sensitivity and specificity and for monads, SPACE performs as good as other tools.

However, based on the submotif notion we define, we implicitly assume that the mismatches are uniformly distributed



**Fig. 8.** Comparison of SPACE and best performing algorithms on four types of organisms. Colour version of this figure is available as Supplementary material online.

in the motif instances. If that is not the case, SPACE may fail to capture these instances, and thus may miss the motif or the regions of the motif that contain these mismatches. On the other hand, in real biological datasets, it seems that mismatches are usually not clustered for most of the motif instances. Hence, SPACE can perform well in most cases.

For future work, we are enhancing our algorithm to handle motifs for which the same gap may have different lengths across different instances by exploring the idea of allowing tolerances in the gap lengths across different instances during the mining process. Other directions include applying our approach for motif finding on *drosophila* s data where the sites maybe overlapping and with fluctuating positions (Makeev, 2003), discovery of motif modules (co-operating binding factors) (GuhaThakurta and Stormo, 2001).

### ACKNOWLEDGEMENTS

The authors would like to thank the reviewers for all their useful and constructive comments. S.M.Y. was supported in part by the funding from the Research Output Prize (Faculty of Engineering) of the University of Hong Kong.

*Conflict of Interest:* none declared.

### REFERENCES

Ao,W. et al. (2001) Environmentally induced foregut remodeling by PHA-4/FoxA and DAF-12/NHR. *Science*, **305**, 1743–1746.  
 Bailey,T. and Elkan,C. (1995) Unsupervised learning of multiple motifs in biopolymers using expectation maximization. *Machine Learning*, **21**, 51–80.  
 Balzi,E. and Goffeau,A. (1995) Yeast multidrug resistance: the PDR network. *J. Bioenerg. Biomembr.*, **27**, 71–6.  
 Becker,B. et al. (1998) A nonameric core sequence is required upstream of the LYS genes of *Saccharomyces cerevisiae* for Lys14p-mediated activation and apparent repression by lysine. *Mol. Microbiol.*, **29**, 151–63.  
 Blanchette,M. and Tompa,M. (2002). Discovery of regulatory elements by a computational method for phylogenetic Footprinting. *Genome Res.*, **12**, 739–748.

- Blanco, E. *et al.* (2006) ABS: a database of annotated regulatory binding sites from orthologous promoters. *Nucleic Acid Res.*, **34**, D63–D67.
- Carvalho, A.M. *et al.* (2003) A highly scalable algorithm for the extraction of cis-regulatory regions. In *Proceedings of the Third Asia-Pacific Bioinformatics Conference (APBC)*, 273–282.
- Dermitzakis, E.T. and Clark, A.G. (2002) Evolution of transcription factor binding sites in Mammalian gene regulatory regions: conservation and turnover. *Mol. Biol. Evol.*, **19**, 1114–1121.
- Eisen, M.B. (2005) All motifs are not created equal: structural properties of transcription factor - DNA interaction and the inference of sequence specificity. *Genome Biol.*, **6**, P7.
- Eskin, E. and Pevzner, P. (2002) Finding composite regulatory patterns in DNA sequences. *Bioinformatics*, **18**, S354–S363.
- Favorov, A.V. *et al.* (2005) A Gibbs sampler for identification of symmetrically structured, spaced DNA motifs with improved estimation of the signal length. *Bioinformatics*, **21**, 2240–2245.
- GuhaThakurta, D. and Stormo, G.D. (2001) Identifying target sites for cooperatively binding factors. *Bioinformatics*, **17**, 608–621.
- Han, J. and Kamber, M. (2000) Data Mining: concepts and techniques. *Morgan Kaufmann*, San Diego, CA. 230–245.
- Han, T.H. *et al.* (1992) Mapping of epidermal growth factor-, serum-, and phorbol ester-responsive sequence elements in the c-jun promoter. *Mol. Cell. Biol.*, **12**, 4472–4477.
- Harbison, C.T. *et al.* (2004) Transcription regulatory code of a eukaryotic genome. *Nature*, **431**, 99–104.
- Hermeking, H. *et al.* (2005) Identification of CDK4 as a target of c-MYC. *Proc. Natl Acad. Sci. USA*, **97**, 2229–2234.
- Hertz, G.Z. and Stormo, G.D. (1999) Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics*, **15**, 563–577.
- Jensen, K.L. *et al.* (2006) A generic motif discovery algorithm for sequential data. *Bioinformatics*, **22**, 21–28.
- Johnston, M. and Carlson, M. (1992) Regulation of carbon and phosphate utilisation. In *Molecular and Cellular Biology of the Yeast Saccharomyces: Gene Expression*, CSHL Press, Woodbury NY, USA 193–281.
- Kato, M. *et al.* (2004) Identifying combinatorial regulation of transcription factors and binding motifs. *Genome Biol.*, **5**, R56.
- Lawrence, C. *et al.* (1993) Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment. *Science*, **199**, 133–154.
- Lenhard, B. *et al.* (2003) Identification of conserved regulatory elements by comparative genome analysis. *J. Biol.*, **2**, 13.
- Liu, X. *et al.* (2001) BioProspector: discovering DNA motifs in upstream regulatory regions of co-expressed genes. In *Proceedings of the Seventh Pacific Symposium of Biocomputing (PSB)*, pp. 127–138.
- Liu, X. and Wulf, P. (2004) Probing ArcA-P modulon of *Escherichia coli* by whole genome transcriptional analysis and sequence recognition profiling. *J. Biol. Chem.*, **279**, 12588–12597.
- Makeev, V.J. *et al.* (2003) Distance preferences in the arrangement of binding motifs and hierarchical levels in organization of transcription regulatory information. *Nucleic Acids Res.*, **31**, 6016–6026.
- Marsan, L. and Sagot, M.-F. (2000) Algorithms for extracting structured motifs using a suffix tree with an application to promoter and regulatory site consensus identification. *J. Comp. Biol.*, **7**, 345–360.
- McGuire, A.M. (1999) A weight matrix for binding recognition by the redox-response regulator ArcA-P of *Escherichia coli*. *Molecular Microbiology*, **32**, 219–221.
- Owen, G. and Zelent, A. (2000) Origins and evolutionary diversification of nuclear receptor superfamily. *Cell Mol. Life. Sci.*, **57**, 809–827.
- Pavesi, G. *et al.* (2001) An algorithm for finding signals of unknown length in DNA sequences. *Bioinformatics*, **17**, S207–S214.
- Peng, C.-H. *et al.* (2006) Identification of degenerate motifs using position restricted selection and hybrid ranking combination. *Nucleic Acids Res.*, **34**, 6379–6391.
- Record, M.T. *et al.* (1996) *Escherichia coli*. RNA polymerase  $\sigma^{70}$  promoters, and the kinetics of the steptranscription initiation. In *Escherichia Coli and Salmonella*, **1**, 792–820.
- Rigoutsos, I. and Floratos, A. (1998) Combinatorial pattern discovery in biological sequences. *Bioinformatics*, **14**, 55–67.
- Schjerling, P. and Holmberg, S. (1996) Comparative amino acid sequence analysis of the C6 zinc cluster family of transcriptional regulators. *Nucleic Acid Research*, **24**, 4599–4607.
- Sinha, S. and Tompa, M. (2000) A statistical method for finding transcription factor binding sites. In *Proceedings of the 8th International Conference on Intelligent Systems for Molecular ISMB-00*, 344–354.
- Svetlov, V.V. and Cooper, T.G. (1995) Compilation and characteristics of dedicated transcription factors in *Saccharomyces cerevisiae*. *Yeast*, **11**, 1439–1484.
- Tavazoie, S. *et al.* (1999) Systematic determination of genetic network architecture. *Nat. Genet.*, **22**, 281–285.
- Thijs, G. *et al.* (2002) A Gibbs sampling method to detect over-represented motifs in the upstream regions of co-expressed genes. *J. Comput. Biol.*, **9**, 447–464.
- Tompa, M. *et al.* (2005) Assessing computational tools for the discovery of transcription factor binding sites. *Nat. Biotechnol.*, **23**, 137–144.
- van Helden, J. (2003) Regulatory sequence analysis tools. *Nucleic Acids Res.*, **31**, 3539–3596.
- van Helden, J. *et al.* (2000) Discovering regulatory elements in non-coding sequences by analysis of spaced dyads. *Nucleic Acids Res.*, **28**, 1808–1818.
- Wasserman, W.W. and Fickett, J.W. (1998) Identification of regulatory regions which confer muscle-specific gene expression. *J. Mol. Biol.*, **278**, 167–181.
- Wei, Z. and Jensen, S.T. (2006) GAME: detecting cis-regulatory elements using a genetic algorithm. *Bioinformatics*, **22**, 1577–1584.
- Werner, T. (1999) Models for prediction and recognition of eukaryotic promoters. *Mamm. Genome*, **10**, 168–175.
- Yagi, H. *et al.* (1995) Regulation of the mouse histone H2A.X gene promoter by the transcription factor E2F and CCAAT binding protein. *J. Biol. Chem.*, **270**, 18759–18765.