Efficient Extraction of Structured Motifs Using Box-links

Alexandra M. Carvalho*
INESC-ID
Rua Alves Redol, 9
1000-029 Lisboa, Portugal
asmc@algos.inesc-id.pt

Ana T. Freitas
IST/INESC-ID
Rua Alves Redol, 9
1000-029 Lisboa, Portugal
atf@inesc-id.pt

Arlindo L. Oliveira
IST/INESC-ID
Rua Alves Redol, 9
1000-029 Lisboa, Portugal
aml@inesc-id.pt

Marie-France Sagot †‡
Inria Rhône-Alpes
Université Claude Bernarde, Lyon I
43 Bd du 11 Novembre 1918
69622 Villeurbanne Cedex, France
Marie-France.Sagot@inria.fr

Abstract

In this paper, we propose a new algorithm for the extraction of repeated motifs that may represent binding-site consensi in genomic sequences. In particular, the algorithm extracts structured motifs, which we define as a collection of highly conserved motifs with pre-specified sizes and spacings between them. This type of motifs is highly relevant in the search for gene regulatory mechanisms since promoter models can be effectively represented by structured motifs.

The algorithm uses factor trees, a variation of suffix trees, and a new data structure, called box-links, to store the information about conserved regions that repeat often in the dataset sequences. The complexity analysis shows a gain over previous algorithms that is exponential on the spacings between boxes.

The application of a prototype implementation of this algorithm to biologically relevant datasets shows the ability of the method to extract relevant consensi. The experimental results also show that this algorithm is much faster than existing ones, sometimes by more than two orders of magnitude.

Keywords Box-link, Suffix tree, Structured motifs, Promoter, Complexity.

1 Introduction

In this large-scale genome sequencing era, the main bottleneck to progress in molecular biology is data analysis. The aim of this analysis is the extraction of different kinds of biological

*Partially supported by FCT grant SFRH/BD/18660/2004 and FCT Project FEDER POSI/SRI/47778/2002 BioGrid

†Visiting Research Fellow at King's College London, UK.

‡Partially supported by CNRS-INRIA-INRA-INSERM action BioInformatique and Wellcome Trust Foundation.

information from sequence data. An important task in this context consists in the detection of regulatory sites in DNA sequences as well as the prediction of the corresponding promoter. Promoter regions can play an important role in gene function and may offer some clues as to the function of completely anonymous proteins. Prediction of the functionality of a promoter may also yield initial indications for gene therapy approaches, while analysis of the combinatorics of their elements are essential for understanding cell development.

An important part of gene regulation is mediated by specific proteins, called the *transcription factors*, which influence the transcription of a particular gene by binding to sequence specific sites on DNA sequence, called *transcription factor binding sites*. Such binding sites are located in *promoter regions*. In prokaryotic organisms, the binding sites are predominantly in the immediate vicinity of the gene, which usually extends about 300 to 600 nucleotides upstream of the transcription start site. However, in eukaryotic organisms the site sequences are often shorter, and can be quite variable and distributed over very large distances. A detailed explanation on possible models for prediction and recognition of eukaryotic promoters can be found in [1].

Given the flexibility of regulatory mechanisms, it is essential to develop methods that are capable to systematically detect different kinds of regulatory signals and to adapt to different promoter models. The literature on the topic of DNA binding site sequence detection is extensive (see [2, 3] for two surveys), and the subject has gained a renewed interest in the last few years, with the sequencing of the genome of vertebrates such as man and mouse.

Up to some five years ago, all methods for detecting DNA binding sites considered each such site individually. These methods therefore looked for motifs composed of a unique element – *single motifs*. This includes pattern-based approaches which allowed for wild cards or a limited number of spacers but not for mutations, apart from [4]. The exceptions to this single motif model were a heuristical approach by Cardon and Stormo [5] which sought for motifs composed of two parts separated by a distance (estimated by the algorithm), a previous work by one of the authors of the present paper [6, 3, 7], and an algorithm which allows for spacers in general [8]. The first method is based on an EM approach to identify sets of words with high relative entropy, the second is a pattern-based approach to motif detection, allowing for a degenerated pattern, while the last is a Fasta-inspired method which seeks for exact short motifs occurring in conserved order along different sequences.

More recently, other methods have appeared to detect motifs composed of two parts, which we henceforward call *boxes*, separated by a spacer, often of fixed-length [9, 10, 11]. Besides considering more complex motifs of a limited type only, with two boxes at most, the algorithms for detecting such motifs are in general naive: they either exhaustively enumerate all possible motifs of two boxes separated by a distance [9], or discover them by crossing the lists of occurrences of single motifs detected in a previous step [10, 11]. In the first case, the method is severely limited in the length of the motifs it can identify. In the second case, detecting motifs with two boxes by crossing the lists of single motifs takes time at least quadratic in the number of such single motifs and their occurrences. To address this problem, the lists for single motifs are trimmed by statistical significance before the crossing operation. However, a motif with two boxes may be statistically significant even though none of the boxes taken individually are. Indeed, one of the main interests in seeking for complex motifs directly lies in this fact. Another well known single motif detection algorithm is MEME [12]. It is, Like Cardon's and Stormo's method, an EM-based algorithm that identifies motifs with high relative entropy, but suffers from the same problem as Eskin *et al.*'s. It identifies significant sets of compatible motifs but works by iteratively building such multiple motifs from single

ones, whose occurrence positions do not contradict the occurrence positions of the single motifs identified. Furthermore, the set of motifs produced must only satisfy compatibility. No constraint, and therefore no statistical value is put on the distances separating them.

This paper is based on the only previous algorithm that is able to identify motifs composed of any number of boxes – *structured motifs* [6]. There are two central problems concerning motifs in sequences: localization and extraction [13]. The goal of the *structured motif localization* is to find the positions in a sequence of the occurrences of a given structured motif [14]. The *structured motif extraction* aims to identify *de novo* structured motifs in a set of input sequences [6]. In this paper, we address the structured motif extraction problem.

The main contribution of this paper is a new exact algorithm to extract multiple binding site consensi from genomic sequences. Like the previous algorithm [6], this new exact algorithm infers all structured motifs which match a minimum number of input sequences. The latter number is called *quorum*. The contents of the boxes, which represent the extracted motifs, are unknown at the start of the process. The algorithm proposed in this paper uses a *factor tree* [15], which is a suffix tree [16, 17, 18, 19] built only up to a certain level, and a new data structure, called *box-links*, to store the information about conserved regions that repeat often in the dataset sequences. The major gain of this new method, over previous approaches for extracting structured motifs, is that the extraction time of the motifs remains independent of the distances between them, leading to an exponential reduction in the worst case analysis. This improvement is very important for the extraction of eukaryotic transcriptional factors binding sites since the promoter model can be very complex with consensus sequences observed over very large and flexible distances.

We do not address in this paper the the evaluation of the statistical significance of the structured motifs found. Work on performing such evaluation for two boxes exists [20] but this remains an open problem for more than two boxes and for more general statistical models. We therefore continue using for such evaluation the same procedure as in the previous algorithm, that is, simulation [3, 7].

In Section 2, we present some previous work that allows a better understanding of the proposed algorithm. In Section 3, we introduce the concept of box-links (Section 3.1) and present the new algorithm for the structured motifs extraction using this new data structure (Section 3.2). In Section 4, we present an algorithm to build box-links. Section 5 presents experimental results that validate the performance and the applicability of this new method.

2 Preliminaries

Algorithms for structured motifs extraction [6] address the extraction of consensus motifs that appear together in a well-ordered and regularly spaced manner. A structured motif can be described as an ordered collection of $p \geq 1$ boxes, a maximum allowed error (substitutions) for each box, and an interval of distance for each pair of consecutive boxes. A suffix tree [16, 17, 18] is used to find such motifs in a set of N input sequences over an alphabet Σ.

To set up the algorithm to extract structured motifs, we have to introduce some notation. A *motif* is an element in Σ^+ for Σ an alphabet. A motif m is said to have an *e-occurrence*, or simply an *occurrence*, in the input sequences, if there is one word u in the input sequences such that the Hamming distance[1] between u and m is less than or equal to e. A motif is

[1]The Hamming distance between two sequences of same length is the minimum number of substitutions to transform one sequence into another.

said to be a *valid motif* if it has an occurrence in at least q input sequences, where q is called the *quorum*. Moreover, a *node-occurrence* of a motif m is represented by (v, e_v) where v is a tree node and $e_v \leq e$ is the Hamming distance between the label of the path from the root to v and m. Finally, a *structured motif* is a pair (m, d) where m is a p-tuple of single motifs $(m_i)_{1 \leq i \leq p}$, denoting the p boxes and d is a $(p-1)$-tuple of triplets $(d_{\min_i}, d_{\max_i}, \delta_i)_{1 \leq i \leq p-1}$, denoting the $p-1$ intervals of distance. The terms $d_{\min_i} \leq d_{\max_i}$ represent a minimum and maximum allowed distance between the parts and δ_i an allowed interval around that distance. When $\delta_i = (d_{\max_i} - d_{\min_i} + 1)/2$, δ_i is omitted. A structured motif (m, d) is said to be a *valid structured motif* if for all $1 \leq i \leq p-1$ and for all occurrences u_i of m_i, there exist occurrences $u_1, \ldots, u_{i-1}, u_{i+1}, \ldots, u_p$ of the single motifs $m_1, \ldots, m_{i-1}, m_{i+1}, \ldots, m_p$ such that: $u_1, \ldots, u_p$ belong to the same input sequence; there exists d_i, with $d_{\min_i} + \delta_i \leq d_i \leq d_{\max_i} - \delta_i$, such that the distance between the end position of u_i and the start position of u_{i+1} in the sequence is in $[d_i - \delta_i, d_i + \delta_i]$; d_i is the same for p-tuples of occurrences present in at least q distinct sequences.

We are now able to describe the algorithm to extract structured motifs [6]. This algorithm is based on a previously published approach to the extraction of single motifs [4]. For clarity of exposition, we assume that all p boxes of a structured motif have the same size k with distances between them over the interval $[d_{min}, d_{max}]$, that each box has a maximum allowed error e and that δ equals $(d_{max} - d_{min} + 1)/2$. The algorithm works as follows. At first a suffix tree $\mathcal{T}$ is built for all input sequences. The suffix tree needs to be modified in order to store at each tree node v a Boolean array of size N, denoted by $Colors_v$ [4], indicating the sequences in the input set that contain the string labeling the path from the root to the tree node v. The structured motif extraction algorithm starts by extracting single motifs m_1 of length k, one at a time. The extraction of single motifs is done by a simple depth-first traversal of the suffix tree of the input sequences [4]. For each node-occurrence v of a first box m_1 (Figure 1a), a jump is made on the suffix tree to the descendants of v situated at distances $[k + d_{min}, k + d_{max}]$ from v (Figure 1b). The content of the Boolean array $Colors$ of the nodes reached at these lower levels is grabbed and carried along the unique path of suffix-links that leads back to a node at level k (Figure 1c). Back at level k, the grabbed information is used to temporarily and partially modify the suffix tree (Figure 1d). The extraction of the second box m_2 of a potentially valid structured motif then proceeds in the same way (Figure 1e). Once the operation of extracting all possible valid motifs $\langle (m_1, m_2, \ldots, m_p), (d_{min}, d_{max}) \rangle$ has ended, the suffix tree is restored to its previous state. The construction of another single motif m_1 follows, and the whole process unwinds in a recursive way until all valid structured motifs are extracted. Figure 1 illustrates the extraction of a structured motif with 2 boxes with same size k.

In terms of complexity, and for p boxes of size k distanced by $[d_{min}, d_{max}]$, the extraction of structured motifs takes $O(N s_p(k, d_{max}) \nu^p(e, k) + N n_{pk+(p-1)d_{max}} \nu^{p-1}(e, k))$ time and $O(N^2 n + N(p-1)n_k)$ space, where n_x is the number of tree nodes at depth x, $s_p(k, d_{max}) = \min\{n_k^p, n_{pk+(p-1)d_{max}}\}$, $\nu(e, k)$ is the number of distinct words that are at a Hamming distance at most e from a k-long word, and n is the average size of an input sequence [6].

3 Structured motifs extraction using box-links

In this section we introduce the main contribution of this paper. A new data structure, called box-link, is used to improve the algorithm presented in Section 2. Intuitively, a box-link stores

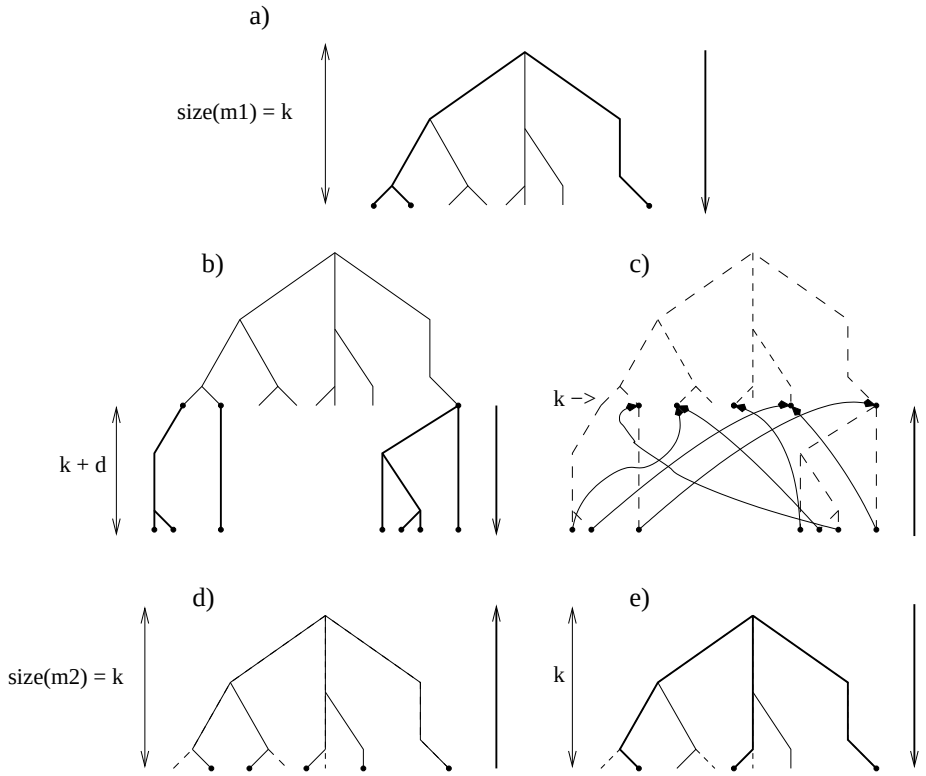

Figure 1: Extracting structured motifs following suffix links.

the information needed to jump from box to box in a structured motif. Its name comes from the fact that it links all p boxes of a possibly valid structured motif. There are two main advantages in the use of this data structure. First and foremost, the information needed to jump from box to box when searching for structured motifs is memorized and can be accessed very fast. Naturally, we pay something with the use of box-links, but as we shall see, both theoretical and experimental analyzes show that they accomplish a substantial improvement. Second, we capitalize on the use of factor trees [15], since there is no need to compute the suffix tree below the size of the maximum box.

3.1 Box-links

In this section, we introduce the new box-link data structure. Its construction from the input sequences is straightforward and it is presented in Section 4.

Suppose we have all p boxes of the same size k with distances between them over the interval $[d_{min}, d_{max}]$. Formally, a box-link can be defined as follows:

Definition 3.1 Let L be the set of leaves at depth k of a k-factor tree $\mathcal{T}$ for a string s and L_k^i denote all possible i-tuples over L. A *box-link of size* i, with $1 \leq i < p$, is a $(i+1)$-tuple in L^{i+1} such that there is a substring s' of s where:

- the length of s' is $ik + (i-1)d$;
- the k-length substring of s' ending at position $jk + (j-1)d$, with $1 \leq j \leq i$, is the path in $\mathcal{T}$ spelled from the root to the j-th leaf of the box-link tuple.

For the sake of simplicity, consider a structured motif constituted by two boxes with size k and distance d between them. Intuitively, there is a box-link from a leaf l_1 to a leaf l_2, if when jumping down the suffix tree from l_1 at depth k (Figure 2a), and taking along the unique path of suffix-links from all children of l_1 at depth $2k + d$, we reach l_2 again at level k (Figure 2b). Clearly, there might be several leaves l_2 fulfilling the previous condition.

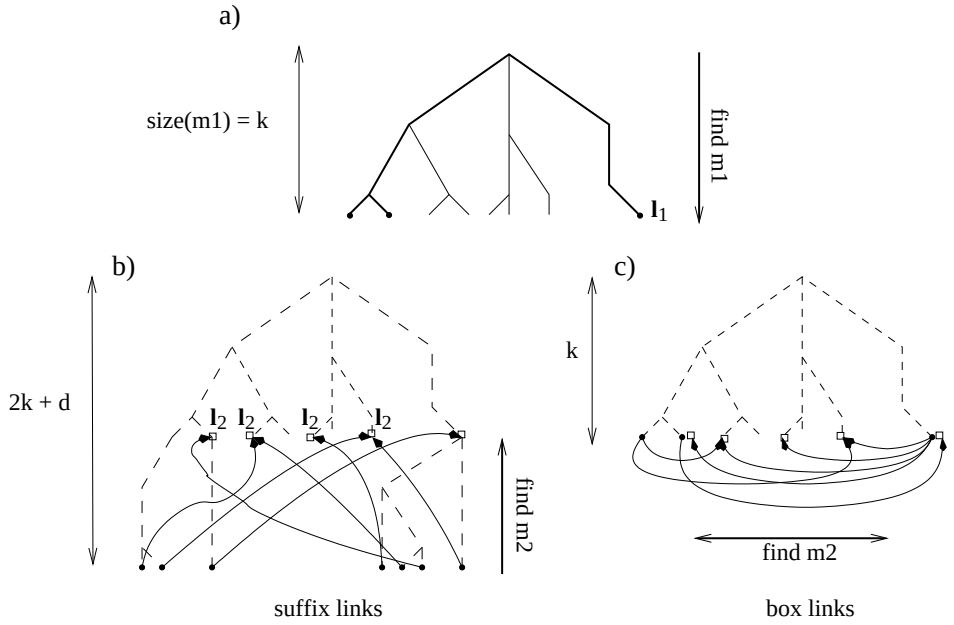

Figure 2: Extracting structured motifs following b) suffix-links and c) box-links.

Figure 2 depicts the differences between the algorithm presented in Section 2 and the proposed one. The information grabbed from level $2k + d$ to level k of the suffix tree is the $Colors$ of the nodes reached at depth $2k + d$. To store all this information we have to endow box-links with a Boolean array, similar to the one associated with tree nodes, defined as:

$$Colors_b[i] = \begin{cases} 1 & \text{if the box-link } b \text{ links boxes of the } i\text{-th input sequence} \\ 0 & \text{otherwise} \end{cases} .$$

3.2 Jumping in the factor tree using box-links

In this section, we describe the algorithm to extract structured motifs, using box-links as a fundamental data structure. We also present its complexity analysis with an exponential time gain in the worst case scenario. This improvement is attained because the time complexity does not depend on the distances between the boxes of the structured motif.

Suppose we have all p boxes of the same size k with distances between them over the interval $[d_{min}, d_{max}]$. The ExtractMotifs algorithm using box-links is very similar to the one proposed with the use of suffix-links [6]. At first a factor tree $\mathcal{T}$ is built, up to the level k, for all input sequences. The factor tree is then modified to store at each node the $Colors$ array, and box-links are added to the leaves of the factor tree. After this pre-processing phase the extraction begins and the pseudocode for the extraction is presented in Algorithm 1. The

Algorithm 1 ExtractMotifs(Motif m, Box i)

1. for each node-occurrence v of m
2. for each leaf z such that there is a box-link $b_{\langle v,z\rangle}$ from v to z
3. put z in $L(i)$
4. if (first time z is reached)
5. $Colors_z = \overrightarrow{0}$
6. put z in $NextEnds$
7. $Colors_z = Colors_z + Colors_{b_{\langle v,z\rangle}}$
8. UpdateTree($\mathcal{T}$,$NextEnds$)
9. for each motif m_i obtained by SpellMotifs traversing $\mathcal{T}$
10. if ($i < p$) ExtractMotifs($m = m_1 \ldots m_i$,$i+1$)
11. else KeepMotif($m = \langle(m_1, \ldots, m_p)$,$(d_{min}, d_{max})\rangle$)
12. RestoreTree($\mathcal{T}$,$L(i)$)

ExtractMotifs algorithm makes use of three functions. First, UpdateTree updates the Boolean arrays from the nodes in $NextEnds$ to the root in the following way: if nodes $\overline{z}$ and $\widehat{z}$ have the same parent z, then $Colors_z = Colors_{\overline{z}} + Colors_{\widehat{z}}$. Any arc from the root that does not have a node in $NextEnds$ is not part of the updated tree, nor are the subtrees rooted at its node in $NextEnds$. Second, RestoreTree restores the Boolean arrays from the nodes in $L(i)$ to the root in the following way: if nodes $\overline{z}$ and $\widehat{z}$ have the same parent z, then $Colors_z = Colors_{\overline{z}} + Colors_{\widehat{z}}$. Any arc from the root is part of the restored tree. Third, KeepMotif stores all information concerning valid motifs.

Next, we establish the time and space complexity for the algorithm at hand. We do not present the proofs because of space restrictions. Moreover, we have to consider $b_p(k, d_{max}) = \min\{n_k^p, n_{pk+(p-1)d_{max}}\}$ to be an upper bound for the total number of $(p-1)$-size box-links.

Proposition 3.2 Algorithm 1 takes $O(Ns_p(k, d_{max})\nu^p(e, k) + Nb_p(k, d_{max})\nu^{p-1}(e, k))$ time and $O(Nb_p(k, d_{max}) + Npn_k)$ space.

This algorithm exhibits an exponential worst case time gain relatively to the previous algorithm to extract structured motifs presented in Section 2. The only difference between complexity expressions appears in the second parcel, concerning the update and restore of the suffix or factor tree, where the new algorithm presents the factor $b_p(k, d_{max})$ whereas the previous algorithm presents the factor $n_{pk+(p-1)d_{max}}$. Observe that in the worst case scenario, the suffix or factor tree is complete and we have $b_p(k, d_{max}) = \min\{|\Sigma|^{pk}, |\Sigma|^{pk+(p-1)d_{max}}\} = |\Sigma|^{pk} < n_{pk+(p-1)d_{max}} = |\Sigma|^{pk+(p-1)d_{max}}$ which reflects an exponential gain of the order $|\Sigma|^{(p-1)d_{max}}$. The major gain of this new method, over previous approaches for extracting structured motifs, is that in the worst case scenario the extraction time of the motifs remains independent of the distances between them.

Few changes to the algorithm are required when one wishes to consider structured motifs composed of boxes with different sizes, or separated by intervals of distance of the type $d \pm \delta$

for some d and a fixed δ. It is easy to adapt the solutions proposed in [6] to take advantage of box-links, but for the sake of space we omit the details.

4 Box-links construction

In this section, we present an algorithm to build box-links. The algorithm takes into account the fact that some box-links may collapse from some box on, by placing them in a same equivalence class, leading to a time complexity negligible in comparison with the time spent extracting structured motifs. Moreover, space requirements do not represent a problem, as we shall see.

Suppose we have all p boxes of the same size k with distances between them over the interval $[d_{min}, d_{max}]$. In order to define the BoxLink algorithm, presented in Algorithm 2, we

Algorithm 2 BoxLink(Boxes p, BoxSize k, BoxDistance d, ListLeaf $list_{leaf}$)

1. for (i from 1 to N)
2. while (size of $list_{leaf_i} \geq pk + (p-1)d_{min}$)
3. $[bl_0]_0 = \text{AddBoxLink}(nil, list_{leaf_i}[0], i)$
4. for (j from 1 to $p-1$)
5. for (g from $(j-1)d_{min}$ to $(j-1)d_{max}$)
6. for (h from d_{min} to d_{max})
7. if (size of $list_{leaf_i} \geq pk + (g + h + (p-j-1)d_{min})$)
8. $[bl_j]_{g+h} += \text{AddBoxLink}([bl_{j-1}]_g, list_{leaf_i}[jk+g+h], i)$
9. remove the first leaf of $list_{leaf_i}$

have to make use of two variables. First, the variable $list_{leaf}$ has the list of all leaves inserted in the factor tree, which can be easily obtained during the factor tree construction [15]. In fact, for the sake of exposition, $list_{leaf}$ can be seen as a family of variables $(list_{leaf_i})_{1 \leq i \leq N}$ (one for each input sequence), where each $list_{leaf_i}$ has average length n (the average length of an input sequence). The j-th leaf of $list_{leaf_i}$ represents the factor tree leaf where it is the j-th k-length factor of the i-th input string. Second, the variable $[bl_j]_g$ represents the equivalence class of j-size box-links that have the sum of all j distances between boxes equal to g. Moreover, we have to set up the function AddBoxLink. AddBoxLink(b,v,i) adds a box-link between an existing $(j-1)$-size box-link b and a leaf v for the i-th input sequence. However, it only creates a new box-link if there is not already a box-link between the box-link b and the node v (merging in this way equivalent box-links). In either way, creating or not a new box-link, the AddBoxLink function sets the Boolean array entry i to 1.

Next, we establish time and space complexity for Algorithm 2. We do not present the proofs because of space restrictions. Recall that $b_p(k, d_{max}) = \min\{n_k^p, n_{pk+(p-1)d_{max}}\}$ is an upper bound for the total number of $(p-1)$-size box-links.

Proposition 4.1 Algorithm 2 takes $O(N^2 n \Delta^2 p^2)$ time and $O(N b_p(k, d_{max}))$ space.

To compare complexity results for time and space, it is important to notice that $b_p(k, d_{max}) \leq Nn$. Hence, the space complexity of the algorithm at hand can also be given by $O(N^2 n)$,

which is less than its time complexity, as expected. Moreover, it is important to notice that time spent building box-links is negligible in comparison to time spent in the extraction presented in Section 3.2.

The algorithm to build box-links requires some changes when one wishes to consider structured motifs composed of boxes with different sizes. In fact, when dealing with boxes of variable length, the label of a tree arc between two nodes may have more than one symbol, since the factor tree is a compact tree, and so it may represent more that one box (of different sizes). This property of factor trees requires box-links to be endowed with some extra structure when dealing with boxes of variable length. A straightforward way to deal with this problem is to store information about the size of the boxes the box-link concerns aside from the input sequence where it occurs. For the sake of space we omit further details.

5 Experimental results

In this section, we present preliminary results obtained using a prototype implementation of the algorithms described in this paper. The structured motifs extraction original paper [6] introduced two algorithms. A C implementation of the first algorithm (not described here) was made available by the authors (SMILEv1.45). For the second algorithm (SMILEv2), the one presented in Section 2, we used a C/C++ implementation. The new algorithm proposed in this paper was implemented on top of the SMILEv2 algorithm augmented with a new data structure to code box-links (RISO).

As test sets we used two groups of sequences. The first group corresponds to a collection of 68 genes that are known to be regulated by zinc cluster factors [21]. The upstream sequences were retrieved from positions -1 to -1000 relative to the ORF start positions. For the dyad analysis of this data, we tested two models. A first model with two boxes with length 3 and distance 11 and a second one with two boxes with length 4 and distance 9. For the first model the algorithm detected, with a 10% quorum, the consensus $CGGn_{11}CCG$ that corresponds to the nucleotides that enter into direct contact with Gal4p. For the second model the consensus $CGGAn_9TCCG$ was detected with highest significance. These consensi, that were classified as statistically significant, are also biologically relevant. The results are presented in Table 1.

# Errors		CPU Times (in seconds)			# extracted
Box 1	Box 2	SMILEv1.45	SMILEv2	RISO	models
1	1	44.72	7.4	0.12	4096
2	2	1612.68	60.71	12.12	65536

Table 1: Extraction of $CGGn_{11}CCG$ and $CGGAn_9TCCG$ from the first dataset.

The second group is composed of three sets of non-coding sequences located between two divergent genes and extracted from the whole genomes of *B. subtilis*, *E. coli* and *H. pylori*. The first of these sets contains 1,062 sequences for a total of 196,736 nucleotides. The second set contains 1,148 sequences and 226,928 nucleotides. The last set contains 308 sequences and 52,100 nucleotides. These three datasets were originally used as test cases for extracting promoter consensi and to test the sequential algorithm [6]. For these datasets we tested a

model with two boxes of size 6 and spacing 17. For the *E. coli* organism, the algorithms were able to detect the consensus $TTGACAn_{17}TATAAT$ given in the literature for the σ^{70} promoter sequence.

# Errors		CPU Times (in seconds)			# extracted
Box 1	Box 2	SMILEv1.45	SMILEv2	RISO	models
1	2	1429.81	1983.41	942.42	11147160

Table 2: Extraction of $TTGACAn_{17}TATAAT$ from the *E. coli* dataset.

Besides biological significant data, we also compared SMILEv1.45 and RISO with synthetic data. The results obtained revealed a similar speedup to the ones presented above. Just for an illustration purpose, an experiment for three boxes of length four distanced by 5..7 and 15..23 is presented in Table 3.

# Errors			CPU Times (in seconds)		# extracted
Box 1	Box 2	Box 3	SMILEv1.45	RISO	models
1	1	2	2141.09	22.86	262131

Table 3: Extraction motifs with three boxes from synthetic data.

By analyzing these results, together with others that we do not present here, we reached some conclusions about the three algorithms. On one hand, SMILEv2 and RISO perform better when the tree is dense, or when the first box of the structured model being extracted is small. In fact, in our experiments, we get to the conclusion that there is a level in the suffix or factor tree from which the tree becomes very sparse. If the size of the first box is smaller than this level, SMILEv2 and RISO achieve much better results than SMILEv1.45. Moreover, if the tree is dense, this level is deeper and we are closer to the worst case scenario where RISO presents an exponential gain over SMILEv2, which by itself presents an exponential gain over SMILEv1.45. This is the case illustrated in Table 1 and Table 3, where the tree becomes sparser at level eight, and we extracted boxes of length three and four. On the other hand, SMILEv1.45 gets better results when we are extracting a structured model with a large first box, or when the input data sequences are poor, as illustrated in Table 2. However, in practice, cases where RISO achieves better results are much more relevant because the input data is richer and the problem is harder.

In what concerns memory, all algorithms performed similarly in our experiments, using about 20Mb. Box-links did not cause memory problems, and in same cases, when boxes were small, the total memory of factor tree and box-links was less than the memory required by the full suffix tree.

6 Conclusion

We presented a new algorithm and data structure for the extraction of repeated motifs in DNA sequences. This work presents several major contributions.

The first significant contribution is the new algorithm for extracting structured motifs, which exhibits an exponential worst case time gain relatively to existing algorithms for ex-

tracting structured motifs. This improvement is obtained because the time complexity does not depend on the distances between the boxes of the structured motif. The only added cost comes from the computation of box-links but this time is negligible in comparison with the time required to perform the extraction of the structured motifs. Moreover, the proposed algorithm only requires the creation of a suffix tree pruned at the depth of the largest box of the structured motif (called a factor tree [15]), saving much space in comparison with the algorithms proposed in [6] that are based on the full suffix tree. On the other hand, the box-links data structure may consume more space, but it is easy to make a trade-off between computing some box-links and having others stored in memory.

Another significant contribution comes from an implementation of the proposed algorithm, that was applied to extract frequent motifs in a few biologically relevant datasets.

The final significant contribution is the presented empirical comparison between our implementation of this algorithm and available implementations of existing algorithms that are capable of detecting multiple motifs composed of any number of boxes. The experimental results show that the new algorithm is much faster than the SMILEv1.45 algorithm [6], in some cases, more than two orders of magnitude faster.

Future work can progress in several directions. From an algorithmic point of view, and in particular in what concerns parallel algorithms, we shall apply the parallelization technique proposed in [22] to the algorithm at hand. This approach will allow the use of even larger datasets which is a major limitation of current solutions. From an implementation point of view, it is worthwhile noticing that the code developed for the prototype can be further optimized by using faster data structures.

Acknowledgments: The authors would like to thank Julien Allali for making available the code of the factor tree implementation.

References

[1] Werner, T.: Models for prediction and recognition of eukaryotic promoters. Mammilian Genome **10** (1999) 168–175

[2] Brazma, A., Jonassen, I., Eidhammer, I., Gilbert, D.: Approaches to the automatic discovery of patterns in biosequences. Journal of Computational Biology **6** (1998) 281–297

[3] Vanet, A., Marsant, L., Sagot, M.F.: Promoter sequences and algorithmical methods for identifying them. Research in Microbiology **150** (1999) 779–799

[4] Sagot, M.F.: Spelling approximate repeated or common motifs using a suffix tree. In: Latin'98. Volume 1380 of Lecture Notes in Computer Science. Spriger-Verlag (1998) 111–127

[5] Cardon, L., Stormo, G.: Expectation Maximization algorithm for identifying protein-binding sites with variable length from unaligned dna fragments. Journal of Molecular Biology **223** (1992) 139–170

[6] Marsan, L., Sagot, M.F.: Algorithms for extracting structured motifs using a suffix tree with an application to promoter and regulatory site consensus identification. Journal of Computational Biology **7** (2000) 345–360

[7] Vanet, A., Marsan, L., Labigne, A., Sagot, M.F.: Inferring regulatory elements from a whole genome. An analysis of the σ^{80} family of promoter signals. J. Mol. Biol. **297** (2000) 335–353

[8] Fraenkel, Y., Mandel, Y., Friedberg, D., Margalit, H.: Identification of common motifs in unaligned dna sequences: application to *escherichia coli lpr* regulon. Computer Applications in Biosciences **11** (1995) 379–387

[9] van Helden, J., Rios, A.F., Collado-Vides, J.: Discovering regulatory elements in non-coding sequences by analysis of spaced dyads. Nucleic Acids Res. **28** (2000) 1808–1818

[10] Eskin, E., Pevzner, P.A.: Finding composite regulatory patterns in dna sequences. In: Intelligent Systems for Molecular Biology (ISMB). (2002)

[11] Eskin, E., Gelfand, M.S., Pevzner, P.A.: Genome-wide analysis of bacterial promoter regions. In: Pacific Symposium on Biocomputing (PSB). (2003)

[12] Bailey, T.L., Elkan, C.: The value of prior knowledge in discovering motifs with meme. In: Intelligent Systems for Molecular Biology (ISMB). (1995)

[13] Crochemore, M., Sagot, M.F.: Motifs insequences: localization and extraction. Handbook of Computational Chemistry (2002) In press.

[14] Morgante, M., Policriti, A., Vitacolonna, N., Zuccolo, A.: Structured motifs search. In: Proceedings of the 8th annual international conference on Computational molecular biology. (2004) In print.

[15] Allali, J., Sagot, M.F.: The at most k-deep factor tree. (2003) Submitted.

[16] Ukkonen, E.: On-line construction of suffix trees. Algorithmica **14** (1995) 249–260

[17] McCreight, E.: A space economical suffix tree construction algorithm. Journal of the ACM **23** (1976) 262–272

[18] Weiner, P.: Linear pattern matching algorithms. In: Proceedings of the 14th IEEE Symposium on Switching and Automata Theory. (1973) 1–11

[19] Gusfield, D.: Algorithms on strings, trees, and sequences. Cambridge University Press (1997)

[20] Robin, S., Daudin, J.J., Richard, H., Sagot, M.F., Schbath, S.: Occurrence probability of structured motifs in random sequences. Journal of Computational Biology **9** (2002) 761–773

[21] van Helden, J., Rios, A., Collado-Vides, J.: Discovering regulatory elements in non-coding sequences by analysis of spaced dyads. Nucleic Acids Research **28** (2000) 1808–1818

[22] Carvalho, A., Freitas, A., Oliveira, A., Sagot, M.F.: A parallel algorithm for the extraction of structured motifs. In: Proceedings of 19th ACM Symposium on Applied Computing. (2004) 147–153