Università di Pisa
Dipartimento di Informatica

Technical Report: TR-06-17

Inference of Approximated Motifs with Conserved Relations

Nadia Pisanti[*‡†] Henry Soldano[*◇] Mathilde Carpentier[◇]
Joel Pothier[◇]

[‡] Dipartimento di Informatica, Università di Pisa, Italy.
[*] Laboratoire d'Informatique de l'Université Paris-Nord, UMR-CNRS 7030, France.
[◇] Atelier de BioInformatique, Université Paris 6, France.
[†] Supported by the ACI IMPBio *Evolrep* project of the French Ministry of Research.

November 9, 2006

Address: via F. Buonarroti 2, 56127 Pisa, Italy. Tel: +39 050 2212700 Fax: +39 050 2212726

Inference of Approximated Motifs with Conserved Relations

Nadia Pisanti*‡† Henry Soldano*◇ Mathilde Carpentier◇
Joel Pothier◇

‡ Dipartimento di Informatica, Università di Pisa, Italy.

* Laboratoire d'Informatique de l'Université Paris-Nord, UMR-CNRS 7030, France.

◇ Atelier de BioInformatique, Université Paris 6, France.

† Supported by the ACI IMPBio *Evolrep* project of the French Ministry of Research.

November 9, 2006

Abstract

In this paper we define a new class of problems that generalizes that of finding repeated motifs. The novelty lies in the addition of constraints on the motifs in terms of relations that must hold between pairs of positions of the motifs. We will hence denote them as *relational motifs*. For this class of problems we give an algorithm that is a suitable extension of the KMR [9] paradigm and, in particular, of the KMRC [15] as it uses a degenerate alphabet. The algorithm contains several improvements with respect to [15] that result especially useful when—as it is required for relational motifs—the inference is made by partially overlapping shorter motifs, rather than concatenating them like in [9]. The efficiency, correctness and completeness of the algorithm is assured by several non-trivial properties that we prove in this paper. Finally, we list some possible applications and we focus on one of them: the study of 3D structures of proteins.

1 Introduction

In this paper we define a new class of problems that extends the traditional inference of repeated motifs. This latter is a well-known problem that consists of finding frequent patterns in a given input text, or, equivalently, patterns shared by several input sequences. This problem has applications in several data mining tasks where data can be represented by a text. For many such applications it is indispensable that a certain degree of approximation is allowed among different occurrences of the same motif. For a survey on combinatorial algorithms for finding approximate repeated motifs, see for example Chapter 5 of [10] or Chapter 4 of [8]. A general observation is that when approximate motifs are sought, the problem becomes computationally critical as there can

be an exponential number of motifs satisfying the required frequency. Such exponentiality is not with respect to input size, but rather on the length of the sought motifs (or in their allowed degree of approximation, somehow often proportional to the length itself), hence the problem is fixed parameter tractable. Nevertheless, this drawback can even lead to unfeasibility and, in "better" cases, to a very noisy output. For this reason there have been attempts in the literature to refine the query in the direction of specifying the *structure* of the motifs [11] or of defining slim *generators* for the complete set of the motifs [12]. The refinement of the former has clear motivations in molecular biology in inferring transcription factors binding sites, while the latter—to the best of our knowledge—still misses a convincing application.

We introduce here a new type of refinement which consists of requiring that also relations between pairs of positions in the motifs are conserved, hence we talk about *relational motifs*. This apparently complicates the problem, but we will exhibit an algorithm that uses a very efficient representation of the motifs and which — thanks to some non-trivial properties we prove in this paper — results in an efficient inference: linear in the input size and "really" fixed parameter tractable. Indeed, refining the query on the motifs reduces the output size and also the explosion of the number of candidates. Moreover, relations allow one to constraint the motifs so that more specific properties are satisfied and thus a more sensitive tool can be conceived. The framework we suggest in this paper is very general, and its solution we exhibit is for the most general case. However, depending upon the specific application, some constraints concerning conservation may be relaxed and hence further efficiency achieved. Indeed, the inference of motifs with relations can find application in many tasks such as music research (detecting scales or just tunes that are in different keys by using the relation that indicates the difference of keys), extracting motifs in trees (where being an ancestor or a father can be explicited by means of relations) or in any sort of structured data such as XML/HTML files (or any other source code), finding geometrical motifs (using points in a plane/space as elements and topological relations among them), studying RNA secondary structures (requiring a Watson-Crick complementarity as relation among fragments of motifs), etc... Each one of these applications has its own peculiarities that can lead to a specific instance of the framework of relational motifs. This specificity is in general driven by a suitable balance between the sensitivity and the efficiency required. In Section 7 we will focus our attention on yet another application in molecular biology, that is finding repeated substructures in 3D protein structures, using as relations the distances between the α-carbons in the protein structure.

We will use two input-defined degenerate alphabets for the description of the motifs (one for the motif elements and one for the relations) thus allowing in general the maximum freedom of approximation. Given that we refer to the paradigm of the KMR algorithm [9], we will have to deal with the degenerate alphabet like in KMRC [15]. In particular, we will restrict our attention to *maximal* motifs for a notion of maximality which is the same as in [15]. The choice of dealing with relational motifs implies the fact that motifs will be inferred by means of an incremental construction by partially overlapping two shorter

motifs. By doing so, we substantially differ from [9] in the same direction as [4] where relations in motifs had been introduced for the first time. In [4], however, several properties were unnoticed and thus unbearable drawbacks introduced. In this paper we will prove some properties that will allow to improve the time and space complexity of [4] by decreasing of an exponential factor the amount of candidates generated at each step that we prove to be redundant.

A preliminary version of part of this paper appeared in [13].

2 Preliminary definitions

Our goal is to find approximate relational motifs on a input text that is a sequence over an alphabet Σ. In this section we formalize the way we express the approximation, and the motifs we want to infer according to this. For a simpler explanation, we start with defining some concepts omitting the relations, that we will integrate in the paradigm later.

Let the input text be a sequence t over the alphabet Σ. We assume that it has length n and we denote this by $|t| = n$. The letter at position p in t is denoted by $t[p]$, and therefore we have that $t = t[1]t[2]\cdots t[n]$ where $t[i] \in \Sigma$ for all $1 \leq i \leq n$.

Definition 1 *Given the alphabet Σ, a* cover *on Σ is a set $G = \{G_1, G_2, ..., G_{|G|}\}$ with $G_i \subseteq \Sigma$ for $1 \leq i \leq |G|$, such that $\cup_i G_i = \Sigma$ and there are no $1 \leq i, j \leq |G|$ with $i \neq j$ such that $G_i \subseteq G_j$. The sets G_i's are said* groups.

The alphabet Σ of the input sequence is implicitly given by means of the sequence itself, while that of the motifs is explicitly given by a cover on Σ defined as above and given as input. The alphabet used to describe the motifs will be that of the groups of such cover, which we will also refer to as degenerate alphabet.

Definition 2 *A k-*pattern *is a k-long sequence on the alphabet of the groups. A k-pattern $x = x[1]x[2]\cdots x[k]$ with $x[i] \in G$ for $1 \leq i \leq k$* occurs *in t at position p if $t[p+i-1] \in x[i]$ for all $1 \leq i \leq k$. In this case p is said to be an* occurrence *of x. We will denote with* extent *the complete set of occurrences of a pattern.*

We are interested in frequent patterns, that are patterns that occur more than a certain number of times. Notice that, due to the degenerate alphabet, different patterns may occur at the same position and that, even more, different patterns may have the very same extent. This is the case when the patterns differ in positions that in the occurrences correspond to letters that belong to the intersections of distinct groups. Hence, given a pattern x, its extent is unique, but this latter may be the extent of other patterns different from x.

Definition 3 *Let k, q be integers and t a sequence on Σ. A* s_motif *of size k for t is a k-pattern that occurs in t at least q times. Given an extent L_I, the* k-motif *I is the set of s_motifs x of size k that have extent L_I. In this case we say that x is an s_motif of I. The parameter q is named* quorum.

When unnecessary or clear from the context, we will omit the k and simply talk about *pattern* and *motif*.

Definition 4 *A k-motif I of t is said to be* maximal *if its extent L_I is not a proper subset of L_J for any other k-motif J. It is* non maximal *otherwise.*

Example 1 *Let us consider the input sequence $\tilde{t} = xbxcxaxbxc$ on $\Sigma = \{a, b, c, x\}$ and the cover $G = \{C_1 = \{a, b\}, C_2 = \{b, c\}, C_3 = \{x\}\}$. Assuming $q = 2$, we have that $C_3C_3C_3$, $C_1C_3C_1$ and $C_1C_3C_2$ are all 3-patterns. Nevertheless, the first never occurs in $\tilde{t}$, the second occurs only at position 6 and thus is not an s_motif either, while the third occurs in 2, 6 and 8 and hence it is an s_motif of size 3. Moreover, the 3-motif with extent $\{2, 8\}$ is not maximal in $\tilde{t}$ because that with extent $\{2, 6, 8\}$ is the extent of another 3-motif; this latter is maximal as well as that with extent $\{1, 5, 7\}$.*

We will say that a k-motif I is a *duplication* if $L_I = L_J$ for any other k-motif J with $J \neq I$. Notice that if I is a duplication of J, then J is a duplication of I, as the relation is symmetrical (and transitive). If a k-motif is maximal and it is a duplication, we will say that it is a *maximal duplication*.

The problem we address is to find the extents of all maximal k-motifs, and it can be formally stated in the following way.

Problem 1 *Finding maximal k-motifs:*

INPUT: The input sequence t, the cover G, and the length k.
OUTPUT: The extents of all maximal k-motifs.

As stated so far, the problem has been solved in [15] by extending the method of [9] to the case of maximal motifs which are approximate in that they are expressed using the degenerate alphabet. Basically, in [15] like in [9], maximal k-motifs are obtained in $O(\log k)$ steps where at each step the length of the motifs is doubled by means of concatenation of shorter motifs, with the difference that in [15] only maximal motifs are kept and hence, in particular, each step is concluded with an exhaustive search of extents included into others in order to detect non-maximal motifs and discard them. The set inclusions detection results to be a sensible bottleneck of the algorithm. In other words, each step of [15] is different from that of [9] as it deals with approximate and maximal motifs, but the two algorithms share the fact that an ℓ-long motif is obtained by a concatenation of two $(\ell/2)$-long motifs that occur in the input sequences at distance $\ell/2$ and in the same relative order. Only if the length k of the sought motifs is not a power of two, there is a final step where the motif of length k are generated by overlapping two motifs of length k' such that $k' < k < 2k'$ and k' is a power of two. We call such a generation an *overlap step*, and the previous ones *concatenation steps*. If the size of an overlap (that is, the length of the

string fragment that the two words share) is o, then we will talk about o-overlap.

The goal of this paper is to further extend the method of [15] to the case in which $O(k)$ steps are overlap steps. In particular, we are interested in o-overlaps with $o \in \Theta(\ell)$ where ℓ is the length of the motifs involved at a generic step. In other words, we infer motifs of growing length where at each step such growth is of a constant factor only instead of doubling the size as in [15]. In this way, the inference of repeated k-motifs requires $\Theta(k)$ overlapping steps while $O(\log k)$ concatenations steps would have sufficed. The need of this apparently useless drawback is motivated by the fact that we introduce relations, as we will show in Section 3.2.

Finally, notice that an obvious variant of the algorithm presented in this paper can solve the problem of finding the maximal motif(s) of maximum length. This can be done by going on incrementing the length until no maximal motif is found, and then possibly finding back the right length with a binary search.

3 Relational motifs

The idea is that in some applications (such as the one we will show in Section 7) it can be useful to extract - not just repeated substrings - but rather substrings that appear approximatively repeated and, moreover, they *mostly* appear somehow arranged in the same *relative* way. For example, the elements can have positions on a plane or space and the relations can be topological relations, or the elements are numbers and the relations are arithmetical binary relations, etc. In molecular biology one can consider RNA secondary structures where the required relation is Watson-Crick complementarity. More in general, our framework allows relations that are independent from the elements themselves. An instance of this that we will show in Section 7, concerns an application to tertiary structures of proteins where we consider the protein's sequence of amino acids as symbols of the sequence, and relations such as the distances between the α-carbons within k-long subsequences in their 3D structure. Notice that in this case the relations are completely independent from the symbol that appear in the sequence (the amino acid) as they only depend from the positions involved. This allows to actually represent and infer 3D structural patterns.

3.1 Definitions

This section will formalize the notion of relational motifs starting with some basic definitions concerning relations and relational motifs that integrate those already given earlier in this paper. In particular we still assume there is an input string $t \in \Sigma^n$ whose p^{th} position is denoted by $t[p]$ and a cover G on Σ. On this string we seek k-motifs which—so far—are sets of strings $x \in G^k$ represented by their complete extents $L_x \subseteq \{1, \ldots, n-k+1\}$. When taking

into account relations, the input string is enriched with the relations that hold between each pair of distinct positions. Notice that there is no need to give relations between positions that are more than k symbols apart as far as we want to infer relational motifs of length k only.

Definition 5 *Let $R = \{r_1, \ldots, r_{|R|}\}$ be the* relations alphabet *and $r \in R$ a* relation*. The* relational input string *t is a n long string on the alphabet Σ where for each pair (p_1, p_2) of positions $1 \leq p_1 < p_2 \leq n$ such that $|p_1 - p_2| \leq k - 1$, it is given the unique (symmetric) relation $r \in R$ that holds between position p_1 and position p_2. We will also denote this with $r(p_1, p_2)$ and with $(p_1, p_2) \in r$.*

Hence, the input size is no longer n, but rather $n \times k$.
Also for the relations we want to allow a certain degree of approximation that makes the framework more general and flexible.

Definition 6 *Let $G_R = \{CR_1, \ldots, CR_{|G_R|}\}$ with $CR_i \subseteq R$ for $1 \leq i \leq |G_R|$ be a* relations cover *on R where the CR_i's are denoted as* relations groups *and none of them is included into another.*

Notice that there is no need to explicitly give the alphabet of the relations as this will be implicit in the relational input sequence. On the other hand, a relations cover such as that we just defined is given as input parameter.
The notion of pattern is also enriched with relations and therefore that of motifs and occurrence as well. Formally:

Definition 7 *A* relational k-pattern *is a k-pattern plus a relation group per each pair of its distinct positions. A relational k-pattern x with relations groups $CR_1, \ldots, CR_{|G_R|}$ (where for each $CR_i \in G_R$ it is indicated the set of pairs (u, v) such that $(u, v) \in r \in CR_i$ for some $1 \leq i \leq |G_R|$) is said to* occur *in t at position p if the pattern x occurs at position p of t and for all pairs $(u, v) \in CR_i$ we have that $r(p + u, p + v)$ for $r \in CR_i$. Finally, given the quorum q, a* relational s-motif *of size k is a relational k-pattern that occurs at least q times, and a* relational k-motif *is the set of relational s-motifs of size k that share an extent.*

Indeed, we will still denote with *extent* the complete set of occurrences of a relational pattern and, moreover, a relational k-motif I_R is said to be *maximal* if its extent L_I is not a proper subset of L_J for any other relational k-motif J_R. Given that there is one (and only one) relation per each pair of positions of the pattern, an extent, together with the length k, denotes again a relational motif that would be unique if it weren't for the degenerate alphabet that holds for relations too.

Example 2 *Let us consider as input sequence our running example of $\tilde{t} =$ xbxcxaxbxc with in addition the alphabet of relations $R = \{r_1, r_2, r_3\}$ with its cover $G_R = \{CR_1 = \{r_1, r_2\}, CR_2 = \{r_2, r_3\}\}$, and such that $(\{(i, i+1) \mid 1 \leq i \leq k-1\} \cup \{(1,4), (2,5), (3,6), (2,6), (4,8)\}) \in r_1$, and $\{(1,3), (3,5), (5,7), (7,9),$*

$(4,7),(7,10),(1,5),(3,7),(5,9),(6,10)\} \in r_2$, *and all other pairs of positions* $1 \leq i,j \leq k$ *are in relation* r_3. *We have that all* 2*-motifs are relational* 2*-motifs because the relations between consecutive positions is always the same and thus definitely conserved. On the other hand, the* 4*-motif with extent* $\{2,6\}$ *is not a relational motif because in its two occurrences the relations between the first and last positions are different and in different groups (because* $(2,5) \in r_1 \in (CR_1 \setminus CR_2)$ *and* $(6,9) \in r_3 \in (CR_2 \setminus CR_1)$*). Moreover, the maximal* 4*-motif with extent* $\{1,5,7\}$ *has two occurrences,* 1 *and* 7*, where the relation between the first and last positions is in* CR_1 *(respectively there are* $(1,4) \in r_1$ *and* $(7,10) \in r_2$*), and again two occurrences, namely* 5 *and* 7*, where such relation is in* CR_2 *because* $(5,8) \in r_3$ *and again* $(7,10) \in r_2$. *Hence, this* 4*-motif corresponds to two distinct relational* 4*-motifs.*

Summing up, the problem we actually aim to solve is the following:

Problem 2 *Finding maximal relational k-motifs:*

INPUT: The input relational sequence t, the cover G, the relations cover G_R*, and the length k.*
OUTPUT: The extents of all maximal relational k-motifs.

3.2 Overlap Steps

In order to take into account relations during the inference phase, each time a candidate k-motif is considered, the conservation of its relations has to be investigated. In this section we count the amount of comparisons that have to be made in order to verify whether relations are conserved in a candidate motif. We do so for three different general ways to perform the inference. The goal of this section is to show that doing overlap steps is the best choice. The analysis we make here ignores the degenerate alphabet; indeed, the fact that motifs are approximated affects the number of comparisons to do, but this affection is independent from that resulting from taking into account relations, as we will see later Section 4. Finally, for the purposes of this section, we ignore the fact that we seek maximal motifs only, because this has no influence in the results we prove here.

Given that each pair of positions of a k-motif has to be in a specific relation, we have that $O(k^2)$ relations have to be checked. Since motifs are built by extending shorter ones, some relations are ensured by the fact that the shorter were already relational motifs, while others have to be checked at the time of the generation of the new motif, and per each one of its occurrences that can be as many as n. Those that have to be checked are the relations involving positions that were not belonging to the same shorter motif involved in the overlap (or in the concatenation or extension). In the literature, there are basically two general ways in which a k-motif can be inferred from shorter ones. Let us consider

the (virtual) trie of all k-patterns that are the candidates whose frequency has to be tested. In order to perform a lossless search for k-motifs, the inference must (virtually) perform a, hopefully partial, visit of this trie. This can be done by attempting to extend the most possible a single candidate at a time and then backtrack to attempt patterns with different prefixes in lexicographical order (i.e. with an *in depth* visit) like in [11]. We will name this *in depth inference*. Another way is to consider at each steps all patterns that are at the same level of the trie (i.e. with an *in width* visit) like in [9]. This is what we call an *in width inference*.

Remark 1 *We point out that, in an input string of length n, the total number of possible occurrences of exact motifs of fixed length are at most n because the extents of distinct exact motifs cannot intersect. This holds independently from the quorum q.*

In a generic intermediate step of an in depth inference, a $(\ell-1)$-long motif is extended by checking the possible conservation of, say, one extra position on its right end and its corresponding relations. The following result counts the maximum number of comparison required in this case in order to check the conservation of the relations for all candidates motifs whose extension is attempted.

Proposition 1 *In a in depth inference of all relational k-motifs in a sequence of length n, there are overall $O(k^3n)$ relations to be checked.*

Proof At a generic step of an in depth inference, the extension with one extra symbol/position of a motif of length ℓ with $1 \leq \ell \leq k-1$ is attempted. The new $(\ell+1)^{th}$ extra position has to check its relations with all positions i for $1 \leq i \leq \ell$, and this has to be done for each occurrence of the motifs. There are at most n distinct occurrences per each fixed length as stated in Remark 1, and hence the relations to be checked are at most as many as $\sum_{\ell=1}^{k-1} n \cdot \sum_{i=1}^{\ell-1} i$, and thus in $O(k^3n)$. •

Let us now make the same counting for the in width inference starting with the KMR case of concatenation steps.

Proposition 2 *In a in width inference of all relational k-motifs in a sequence of length n that makes use of concatenations steps, there are overall $O(k^2n)$ relations to be checked.*

Proof At each step two motifs are concatenated in order to form a new one of double length. There are only $O(\log k)$ such steps to perform, and at each step i, for $1 \leq i \leq (\log k)-1$, we have two motifs of length 2^i that are concatenated; in this case relations between pairs of positions belonging to the two distinct sub-motifs have to be checked, which makes $(2^i)^2$ comparison per each one of the occurrences of the new motif. We have seen in Remark 1 that at a given step,

since the length is fixed, there are overall (for all the motifs that are being generated) at most n positions for which the check has to be done. Therefore, the result is $\sum_{i=1}^{(\log_2 k)-1}(n \cdot 2^{2i}) = \sum_{i=1}^{(\log_2 k)-1}(n \cdot 4^i) = n \cdot 4^{\log_2 k} = n \cdot (2^{\log_2 k} \cdot 2^{\log_2 k})$ and thus we have $O(k^2 n)$ relations that are checked. •

Notice that the result of Proposition 2 holds also when a constant number of overlap steps replace as many concatenations, and when a non constant number of concatenations are replaced by o-overlap steps with $o \in O(1)$. The following result, instead, shows what happens with an *in width* construction that performs a non constant number of o-overlap steps where o is not fixed.

Proposition 3 *In a in width inference of all relational k-motifs in a sequence of length n that makes use of $O(k)$ overlap steps, there are overall $O(kn)$ relations to be checked.*

Proof Assume we perform a total of $O(k)$ $(\ell - d)$-overlap steps of two ℓ-motifs where d is in $O(1)$. We would have, at step i for $1 \leq i \leq k/d$, two $(\ell_i - d)$-motifs that are merged and only the relation between one of the d positions at the extreme right and one of those at the extreme left of the new motif has to be checked, and again for all its occurrences. Given that by Remark 1, these are at most n in total for the whole step, we have that the relations to be checked are at most $\sum_{i=1}^{k/d} n \cdot d^2 = ndk \in O(n \cdot k)$. •

As a result of Propositions 1, 2, and 3, we have that when inferring relational k-motifs, performing $O(k)$ overlap steps is the best choice. Hence, in next sections we will focus on solving the problem of finding all maximal k-motifs by means of overlap steps only.

Performing overlap steps requires some care because an overlap step can overgenerate non maximal motifs. In particular, we will see in Section 4.3 that in the case of maximal approximate motifs, an overlap step of two maximal motifs can generate extents that do not correspond to any motif and that will be detected and discarded as they are properly included into extents of motifs that are generated. This redundant generation causes extra work in the already costly phase of the detection of non maximal motifs. We will characterize this class of extents and show that, fortunately, there is a way to avoid generating them that will be shown in Section 5 and that makes use of some non trivial properties we prove in this paper. This drawback was unnoticed in [15], but there it was not so relevant because there was only one overlap step. In the case of a series of overlap steps as we have here, the absence of this optimization could result catastrophic for several interesting applications.

4 Properties of (Maximal) k-Motifs

4.1 On the cardinality of maximal k-motifs

In this section we prove some properties of maximal motifs that will result useful to set an upper bound on the cardinality of candidate motifs we will have to deal with. Again, we will first exhibit results and examples omitting relations, and we will later integrate them and consequently extend the results.

Let us start by reminding that several s-motifs may have the same extent due to the adoption of the degenerate alphabet, and that we will implicitly represent them all with a motif that is actually their extent. Formally, for the input sequence t, a cover G, and a length k, the extent L represents the following set of patterns of length k (that is a k-motifs if $|L| \geq q$ where q is the quorum):

$$\{x = x[1] \dots x[k] \mid x[i] \in G \text{ such that } t[p+i-1] \in x[i] \; \forall \; 1 \leq i \leq k \; \forall \; p \in L\}.$$

Example 3 *In our running example $\tilde{t} = xbxcxaxbxc$ with $G = \{C_1 = \{a, b\}, C_2 = \{b, c\}, C_3 = \{x\}\}$ we have that for $k=3$ the extent $\{2, 8\}$ (the substring bxc occurs at both positions) represents both $C_1C_3C_2$ and $C_2C_3C_2$.*

Independently from the input sequence and the quorum, the set of distinct patterns of length k can theoretically be as big as the set of different k-long words on the alphabet G, which has size $|G|^k$. In the following example we show a sequence where this is the case.

Example 4 *Let $\overline{\sigma} \in \Sigma$ occur in all groups of G. In the input sequence $\overline{\sigma}^n$ every string in G^k is an s-motif of size k for $1 \leq k \leq n-1$ (this holds for any quorum $1 \leq q \leq n-k+1$).*

Hence, the upper bound happens to be tight. And indeed, although an input sequence such as $\overline{\sigma}^n$ above is quite improbable, in practical cases an explicit representation of all motifs of a given length is unfeasible. On the other hand, observe that the exponential number of k-motifs shown above can be represented by an unique extent $L = \{1, 2, \dots, n-k+1\}$, that is, in linear space. This is a first intuitive motivation of why our algorithm actually deals with extents only. In fact, the above mentioned motifs of the sequence $\overline{\sigma}^n$ can all be represented by an unique extent because they are all maximal duplications of each other. It is easy to observe that, when representing maximal motifs by means of their extents only, duplications are clearly a redundant information.

Unfortunately, although a single extent can represent an exponential number of k-motifs, keeping only the extents does not suffice to avoid the exponential upper bound, not even if we restrict to maximal and non duplicated motifs. Indeed, we give below an example where we exhibit $|G|^k$ maximal and non duplicated (hence with different extents) k-motifs in a string of length $n = |G| \cdot k$.

Example 5 *Let the cover be* $G = \{G_1, G_2, \ldots, G_{|G|}\}$ *with* $G_i = \{\sigma_i, \overline{\sigma}\}$ *for* $1 \leq i \leq |G|$ *(hence* $|\Sigma| = |G| + 1$*). Let us consider the input sequence* $\tilde{\sigma}' = \overline{\sigma}^k \sigma_1 \overline{\sigma}^{k-1} \sigma_2 \overline{\sigma}^{k-1} \cdots \sigma_i \overline{\sigma}^{k-1} \cdots \sigma_{|G|} \overline{\sigma}^{k-1}$ *where each* σ_i *occurs at position* $ik + i$*. Each string* $x \in G^k$ *occurs at position* 1*, and in* k *more positions. Namely, for each* $1 \leq j \leq k$*,* x *occurs in position* $ik + i - j + 1$ *for* $x[j] = G_i$*. Since each* x *has exactly* $k + 1$ *occurrences, none of them can be non maximal. Moreover, given that for different* x *there are different occurrences, then they cannot be duplications of each other. Given that there are* $|G|^k$ *such* x*'s that are* k*-motifs in* $\tilde{\sigma}'$*, then there can be as many as* $|G|^k$ *maximal and non duplicated (hence with different extents)* k*-motifs in a string of length* $n = |G| \cdot k$*, (for any quorum* $1 \leq q \leq k + 1$*).*

Besides the theoretical possibility shown in the example above, we have that in practical applications we are fortunately very far from such worst case, as we will show in Section 7. However, the example points out the crucial role that the cover G, which indicates how much the motif can be approximated, plays in the possible explosion of the number of candidates. Indeed, if exact motifs are sought, then G should trivially coincide with Σ (and it is actually useless to talk about groups). If $G \neq \Sigma$ then we are allowing an approximation in the way the motifs match their occurrences. We now formalize the degree of such approximation.

Definition 8 *Given a* cover $G = \{G_1, G_2, ..., G_{|G|}\}$ *on* Σ*, the* degeneracy g *of* G *is the maximum number of distinct groups to which a same* $\sigma \in \Sigma$ *belongs to.*

In other words, g measures indeed how much *degenerate* is the alphabet of the motifs. For exact motifs we have $G = \Sigma$ and hence $g = 1$, but when an approximation is sought, we have in general $g > 1$ which is somehow a measure of the degree of such approximation. In theory, g can be as big as $|G|$ like in Example 5. Should this be the case, the output motifs would not be significant (and too many). Hence, in practical cases it will not be the case, and this is the reason why the upper bound of Example 5 is not met in practice. Given that we deal with a degenerate alphabet like [15], it can be useful to view the upper bound on the number of k-motifs also in terms of g. In [15] it is proved the following[1].

Proposition 4 *In an input sequence in* Σ^n*, given a cover* G *of* Σ *having degeneracy* g*, for a fixed* k *the total size of the extents of all the* k*-motifs is at most* $min(|G|^k, ng^k)$*.*

In Section 3.2 we have counted the number of relations to be checked ignoring the degenerate alphabet. If we use the upper bound of Proposition 4 instead of that of Remark 1, the results of Propositions 1,2, and 3 can trivially be extended to the case of approximate motifs obtaining new upper bounds where instead of n we have ng^k.

[1] In [15] the result is stated for maximal motifs. However the very same proof works for motifs in general.

4.2 Compositionality of maximal motifs

Since we infer motifs of growing length, it is useful to know that at each step we only need to store maximal motifs because these are enough to produce longer ones. This is possible thanks to the following result.

Lemma 1 *Each maximal k-motif I has an s-motif m whose ℓ-long prefix and ℓ-long suffix ($\forall\ 0 < \ell < k$) are s-motifs of maximal ℓ-motifs.*

Proof Let I be any maximal k-motif with extent L_I, m any of its s-motifs, and let I_p be the prefix of length ℓ of m. It must be that $L_I \subseteq L_{I_p}$ because a proper suffix of any string occurs at least wherever the string does. If I_p is maximal then we are done. If this is not, then it is because there exists another maximal ℓ-motif I'_p such that $L_{I_p} \subsetneq L_{I'_p}$. Since $L_I \subseteq L_{I_p}$, then we have that $L_I \subsetneq L_{I'_p}$, and hence I'_p is a prefix of another s-motif of I (because it occurs wherever I does). Given that by hypothesis I'_p is also maximal, we have that it is what we are looking for. The very same proof can be done for suffixes of I and shifting the corresponding extents, and hence the result is proved. •

Notice that the result of Lemma 1 actually holds for any substring and not just for prefixes and suffixes as the proof does not depend at all from the fact that the substring is a prefix or a suffix.
Given an extent L and an integer d, we denote with $L + d$ the set $\{x + d \mid \forall x \in L\}$. Lemma 1 has the following consequence.

Theorem 1 *The extents of all maximal k-motifs can be computed from the extents of maximal ℓ-motifs for a fixed ℓ such that $k/2 \leq \ell < k$.*

Proof In order to obtain the extent of a maximal k-motif, it suffices to take the extents L_p and L_s of the maximal ℓ-motifs which are its maximal prefix and suffix according to Lemma 1, and compute $L_p \cap (L_s + \ell - k)$. •

As a consequence, the set of *all* the extents of maximal motifs of a fixed length ℓ is sufficient to generate any (hence possibly all of them) maximal motif of length $\ell + d$ provided $\ell > d$. Therefore, we have that in our incremental construction of motifs, at each intermediate step we only need to keep extents of maximal ℓ-motifs in order to generate longer ones up to the required length k.

4.3 Pseudo-motifs

We now show that, even when dealing with extents only and with maximal motifs of fixed length, in general an overlap of two ℓ-motifs can generate quite more than *just* $(\ell + d)$-motifs. We will show why, and also that our algorithm avoids this drawback. Let us start again with a simple example that anticipates the definition.

Example 6 *Let us consider again the running example* $\tilde{t} = xbxcxaxbxc$, $q = 2$, *and* $G = \{C_1, C_2, C_3\}$ *with* $C_1 = \{a, b\}$, $C_2 = \{b, c\}$ *and* $C_3 = \{x\}$. *Let us consider the extent* $\{1, 7\}$ *and length* $k = 3$, *corresponding to the substring* xbx. *This latter is repeated* 2 *times as requested by the quorum and it corresponds to* $C_3(C_1 \cap C_2)C_3$, *which does not match our definition of pattern. Notice that its extent is different from that of* $C_3C_1C_3$ *(that is* $\{1, 5, 7\}$*) and* $C_3C_2C_3$ *(that is* $\{1, 3, 7\}$*) that are both maximal* 3*-motifs. On the other hand,* $(C_1 \cap C_2)C_3C_2$, *which also occurs twice (at* 2 *and at* 8*) and it is not a* k*-pattern, has the same occurrences as the s-motif* $C_2C_3C_2$.

Definition 9 *A* k-pseudo-pattern *is a* k*-long sequence on the alphabet of the subsets of the groups whose extent is not the extent of a* k*-pattern. We name it a* k-pseudo-motif *if it occurs at least* q *times and we name* pseudo-extent *its complete list of occurrences.*

In Example 6 for $k = 3$ we have that $\{1, 7\}$ is a pseudo-extent for the pseudo-motif $C_3(C_1 \cap C_2)C_3$ while $(C_1 \cap C_2)C_3C_2$ is not a pseudo-motif and thus $\{2, 8\}$ is not a pseudo-extent because $\{2, 8\}$ is also the extent of the motif $C_2C_3C_2$. Our concern on pseudo-motifs is motivated by the fact that, given a cover G, there can be $O(2^{|G|k})$ distinct pseudo-motifs of length k because there are as many pseudo-patterns as the number of distinct k-long strings on the alphabet of the subsets of G which are not k-patterns, that is $2^{|G|k} - |G|^k$. Notice, however, that a pseudo-extent can never be an extent of a maximal motif because it is always included into the extent of a k-motif. Namely, if the pseudo-motif is, say, $x = C_1 \cdots (C_i \cap C_j) \cdots C_k$ with extent L, then by definition the k-motif $m = C_1 \cdots C_i \cdots C_k$ has an extent which is different from L and it must necessarily include it because m occurs wherever x does. Hence, due to Theorem 1, pseudo-motifs are not necessary to generate longer maximal motifs. On the other hand, the overlap of two maximal motifs can generate a pseudo-motif, as shown in the following example.

Example 7 *In our running example* $L_1 = \{2, 6, 8\}$, $L_2 = \{2, 4, 8\}$, $L_3 = \{1, 5, 7\}$, *and* $L_4 = \{1, 3, 7, 9\}$ *are the extents of maximal* 2*-motif. If we perform a* 1*-overlap of* m_3 *and* m_2, *we obtain the extent* $\{1, 7\}$ *corresponding exactly to the pseudo-motif exhibited in Example 6. The same happens overlapping* m_4 *and* m_1.

Hence, when overlapping maximal motifs we can generate pseudo-motifs. And not only generating pseudo-motifs would be useless, but they would even introduce a serious drawback on the performance of the method. Indeed, given how many the pseudo-motifs can be (and how many they *are* in practice as we shall see in Section 7), generating them all at each step postponing their detection and elimination to the exhaustive search of included extents would result very inefficient, and mostly unfeasible. More precisely let us suppose that, in the worst case, we have computed the ng^k maximal k long motifs, and that we compute the $k + d$ long motifs using a $k - d$-overlap step. This will result in possibly generating ng^{2k} among motifs and pseudomotifs. As there

cannot be more than ng^{k+d} $k+d$ long motifs, the rest, i.e. $ng^k(g^k - g^d)$ are pseudomotifs. Note that if $k = d$, i.e we make concatenation steps, then there are no pseudomotifs at all. We will see in Section 5 a necessary condition on motifs that will allow us to avoid to generate pseudo-motifs.

4.4 Properties of relational motifs

In this section we extend to the case of relational motifs all the definitions and properties we have given in Sections 4.1, 4.2, and 4.3.

Similarly to the case of the cover G on the alphabet Σ, a *relations degeneracy* g_R notion exists on G_R (defined in the obvious way analogously to Definition 8), and this represents the degree of approximation on the relations in the very same way as g does on the symbols' alphabet Σ.

As we have seen in Example 2, in general to an extent X of a non relational motif may correspond several distinct extents X_i's of relational maximal motifs (being them different subsets of X). This can be the case when in different occurrences hold different relations. Moreover, the higher g_R, and higher is the theoretical possibility that the X_i's can even overlap, giving rise to a further combinatorial explosion of their number. Should these extents be at least as large as q and maximal, we have to retain them all. Therefore, we have to review the upper bound given in Proposition 4 in order to take into account (approximate) relations as well. We point out that what we are seeking is not simply the maximum number of motifs of fixed length, but rather an upper bound of the total amount, over all the extents of relational ℓ-motifs, of text positions that appear in these extents. This will result in the maximum amount of data we have to store at a generic step of the algorithm.

Theorem 2 *Given a length ℓ, a cover G (resp. G_R) with degeneracy g (resp. g_R) for the alphabet Σ (resp. R), in a given relational input sequence of length n, the total size of all extents of relational ℓ-motifs is at most $n(g^{\ell}.g_R^{\ell(\ell-1)/2})$.*

Proof Consider any position x on the input sequence s. We have that each letter can be at most in as many as g groups of the cover G, and thus at position x can start occurrences of at most g^{ℓ} distinct motifs. Moreover, at the same position x there can start a relation motif with its $\ell(\ell-1)/2$ relations, each one being in at most g_R distinct relational groups; hence at x occur at most $g_R^{\ell(\ell-1)/2}$ relational motifs. Since there are less than n possible positions x, the resulting upper bound is $n(g^{\ell}.g_R^{\ell(\ell-1)/2})$. •

Again, this is the only theoretical upper bound we can give, but it is far from being tight in practical cases, as we shall see in Section 7.

The compositionality of maximal motifs holds also for the relational case, as the proofs of Lemma 1 and Theorem 1 can straightforwardly be extended to

prove the following.

Theorem 3 *The extents of all maximal relational k-motifs can be computed from the extents of maximal relational ℓ-motifs for $k/2 \leq \ell < k$.*

Finally, also the definition of relational pseudo-motif is a natural extension of Definition 9. Namely, a *relational k-pseudo-pattern* is a k-long sequence on the alphabet of the subsets of the groups in G, with a subset of the groups in G_R per each pair of positions $1 \leq p_1 < p_2 \leq k$, whose extent is not the extent of a relational k-pattern. A relational k-pseudo-pattern is a relational *k-pseudo-motif* if it occurs at least q times. We omit examples of relational pseudo-motifs as the notation can result heavy and, however, the concept is the very same as that shown in Example 6. The overlap of two relational motifs can generate a relational pseudo-motif (this can be seen adding any relation between the two positions of the motifs of Example 7). Finally, notice that by definition, also relational pseudo-motifs cannot be maximal. We omit the counting of how many relational pseudo-motifs there can be in theory, as we shall see in Section 7 how many occurrences of them we have in practice that we actually avoid to generate as we prove in the next section.

5 The algorithm

5.1 The idea

Once again, we will begin with the simple case of non-relational motifs, and we will point out later in Section 6 the peculiarities of the algorithm that guarantee conserved relations as well.

As we have anticipated earlier, our algorithm performs an incremental inference of maximal motifs of growing length, from short ones to those of the required length, avoiding an explicit enumeration of all motifs. Indeed, we only deal with their extents, resulting in a more compact and efficient representation. In this way, we sensibly decrease the phenomenon of the combinatorial explosion of candidates. Their extents contain all the information we need for filtering maximal motifs and generating longer ones by overlapping them. Only a limited amount of additional information per each maximal extent will allow us, as we will show, to avoid generating pseudo-motifs.

Our algorithm infers maximal k-motifs by incrementally extending maximal motifs by means of pairwise overlaps, until the length k is reached. Roughly, for a given constant d, it performs $O(k/d)$ steps where in each one of them pairs of maximal ℓ-motifs undergo a $(\ell - d)$-overlap starting the set of all maximal ℓ_0-motifs, with ℓ_0 being the smallest power of two greater than d (as no $(\ell - d)$-overlap would be possible for $\ell < d$). Hence, there is a first phase of $O(\log d)$ steps where maximal motifs of growing length are generated with a constant number of concatenations steps as in [15], until the length $\ell_0 > d$ is obtained.

In a second phase, there are $O(k/d)$ steps where in each one of them pairs of maximal ℓ-motifs undergo a $(\ell-d)$-overlap and generate $(\ell+d)$-motifs. Each one of these steps is concluded by a detection of non maximal motifs that are then discarded. Finally, the very last overlap step might possibly involve a $d' < d$ in case $k - \ell_0$ is not a multiple of d. The parameter d will be input defined and its choice will actually depend upon the application. More details will be given in Section 6. In the next section we describe more formally the algorithm whose correctness and completeness is partly due to Theorem 1, but also to further results that will be proved in Section 5.3.

5.2 Pseudocode

Let us now describe in a more detailed way the $O(k/d)$ overlap steps. At step i ($i \geq 0$) we have the extents of all maximal ℓ_i-motifs with $\ell_i = \ell_0 + id$, with which we:

(i) Perform all possible pairwise $(\ell_i - d)$-overlaps of two ℓ_i-motifs, computing the extents of the resulting ℓ_{i+d}-motifs and storing from which pair of ℓ_i-motifs they have been obtained.
(ii) Keep only those whose extents have size at least q.
(iii) Eliminate non-maximal and duplicated extents.

After these three steps we are left with all maximal and non duplicated ℓ_{i+1}-motifs. This is iterated as long as $\ell_i < k$. After that, if $\ell_i = k$ then we have completed the task, and otherwise we perform a final $(2\ell_i - k)$-overlap. We now describe how we intend to minimize the amount of extents generated at step (i) and thus also to speed up the filtering of step (ii), and especially of step (iii) which otherwise would be an unbearable bottleneck. The idea is that for each ordered pair I and J of maximal ℓ_i-motifs the extent of the ℓ_{i+d} motif obtained by overlapping I and J is computed, and the fact that I is its prefix and J its suffix is stored. Later on, whenever a motif X' is discarded in phase (iii) because its extent is included into that of X, X' is eliminated and X adds the prefix(es) and suffix(es) of X' to its. If this is the case, we say that X *inherits* X'. This storage of data about which maximal prefixes and suffixes a motif comes from, and their inheritance for eliminated motifs is motivated by the fact that actually the generation of a new motif will be conditioned by whether or not a simple property concerning this data holds. This condition, that we will refer to as *prefix-suffix condition*, will actually allow us to be guaranteed not to generate any pseudo-motif, as we will see in next section.

At step i an ℓ_i-motif I is described by the following data: the identifier $\#I$, the extent L_I, and a pair (P_I, S_I) of lists indicating the set P_I of prefixes in terms of identifiers used in step $i-1$ (omitting the $\#$), and the set S_I of suffixes in terms of identifiers used in step $i-1$. For an efficient computation and for ease of notation, we will also make use, at step i, of a vector V_i of length n such that $V_i[p] = \{\#I \mid p \in L_I \textit{ at step } i\}$. The algorithm is the following.

// INITIAL PHASE //
1. Create an identifier for each $G_i \in G$ occurring in t and compute its extent;
2. Compute V_1;
3. $\ell_0 := 1$;
4. **while** $\ell_0 \leq d$ **do begin**
5. **for each** maximal ℓ_0-motif I **do for each** $x \in L_I$ **do**
6. **for each** $\#J \in V_1[x + \ell_0]$ **do**
7. $L_{IJ} := L_{IJ} \cup x$;
8. Eliminate non-maximal extents and duplications;
9. $\ell_0 := 2\ell_0$; **end**

// OVERLAP PHASE //

$i := 0$;

10. **repeat**
begin
11. Compute V_i;
12. **for each** maximal ℓ_i-motif I **do for each** $x \in L_I$ **do**
13. **for each** $\#J \in V_i[x + d]$ **do**
14. **if** $S_I \cap P_J \neq \emptyset$ **then begin** $L_{IJ} := L_{IJ} \cup x$; $P_{IJ} := I$; $S_{IJ} := J$ **end**;
15. Detect and eliminate extents below quorum, non-maximal extents and duplications;
16. **for each** eliminated non-maximal or duplicated I' **do**
begin
17. choose <u>one</u> I such that $L_{I'} \subseteq L_I$ with I maximal;
18. $P_I := P_I \cup P_{I'}$; $S_I := S_I \cup S_{I'}$
end;
19. $i := i + 1$; $\ell_i := \ell_i + d$
end
20. **until** $\ell_i > k - d$;

// FINAL STEP //

21. **if** $\ell_i < k$ **then** SAME AS LINES $11 - 15$ WITH $d = k - \ell_i$;

Notice that the pseudocode could be written in a much more compact way grouping the three phases into a unique cycle parametrizing the size of the overlap. We chose to display it as it is for ease of exposition. We denote with L_{IJ} the set $L_I \cap (L_J - d)$ which is the extent which is possibly generated at lines 12-14 by overlapping the two maximal motifs I and J (in fact, lines 12-14 are executed for each $x \in (L_I \cap (L_J - d))$). The condition of line 14 is the prefix-suffix condition. Although not explicitly processed (due to complexity reasons), it is clear that a motif IJ obtained by an $(\ell - d)$-overlap of I and J inherits their composition in the following way. If I (resp. J) was a duplication, it definitely represents several s-motifs; let $G_1 \dots G_d \dots G_\ell$ (resp. $G'_1 \dots G'_{\ell-d} \dots G'_\ell$) be *any* s_motifs of I (resp. J). We also denote this with $I[i] = G_i$ (resp. $J[i] = G'_i$). Then we have that $IJ = G_1 \dots G_d(G_{d+1} \cap G'_1) \dots (G_\ell \cap G'_{\ell-d})G'_{\ell-d+1} \dots G'_\ell$. Therefore, IJ will be a motif only if for all such s_motifs of I and J we will have that for all $1 \leq d \leq \ell - d$ the intersection $(G_{d+i} \cap G'_i)$ restricted to the set $L_{IJ} + d + i$ is equal to G_{d+i} or to G'_i. In other words, definitely $IJ[i] \in G$ for $1 \leq i \leq d$ and $(\ell + 1) \leq i \leq (\ell + d)$, but for all positions where the occurrences

of I and J overlap, whether IJ is a motif is in general an open question whose answer is relevant in terms of complexity issues and it is addressed in the next section.

5.3 Correctness and Completeness

We here prove that the algorithm we just introduced is correct (that is, it outputs *only* maximal k-motifs) and complete (it outputs *all* of them). Correctness requires that only k-long motifs are output and among them only maximal ones. The latter is guaranteed by line 15 of the pseudocode where non maximal motifs are discarded. The former comes directly from the condition of line 20 and the setting of d at line 21. Completeness would be a direct consequence of Theorem 1 should not be for the prefix-suffix condition $S_I \cap P_J \neq \emptyset$. In the remaining of this section we show that this condition, together with the settings of lines 16-18, does not affect the completeness of our method. To this purpose, in particular, we need to show that this condition does not discard any maximal and non duplicated motif, which is proved by the following theorem.

Given an ℓ-motif I with extent L_I in an input sequence s, we denote with $I[p]$ the set of groups that occur at position p of I. That is, $I[p] = \{g \in G \mid s[x+p] \in g \ \forall x \in L_I\}$.

Theorem 4 *Let q be the quorum and let I and J be maximal ℓ-motifs such that $S_I \cap P_J = \emptyset$ and $|L_{IJ}| \geq q$ with $L_{IJ} = L_I \cap (L_J - d)$. Then we have that L_{IJ} is either a pseudo-extent or a duplication for length $\ell + d$.*

Proof Given that $S_I \cap P_J = \emptyset$, we must have that per each pair of $(\ell - d)$-motifs SI, PJ such that $SI \in S_I$ and $PJ \in P_J$ we have that $SI \neq PJ$. This means that there must exist $1 \leq p \leq \ell - d$ such that $SI[p] \neq PJ[p]$ and thus such that $I[p + d] \cap J[p] = \emptyset$ (because $I[p + d] \subseteq \cup_{SI \in S_I} SI[p]$ and $J[p] \subseteq \cup_{PJ \in P_J} PJ[p]$). Let us now consider position $p + d$ of the candidate $(\ell + d)$-motif IJ, that is $IJ[p + d]$. We have that no group $g' \in I[p + d]$ can be in $IJ[p+d]$ because $L_{IJ} \subseteq (L_J - d)$, and thus g' would belong to $I[p+d] \cap J[p]$ which is empty by hypothesis. Similarly, no group of $J[p]$ can be in $IJ[p + d]$, otherwise it would also be in $I[p + d]$ (because $L_{IJ} \subseteq L_I$), contradicting again that $I[p + d] \cap J[p] = \emptyset$. Therefore, L_{IJ} is either the extent of a pseudo-motif, or the extent of a motif that in $IJ[p + d]$ has one (or more) group(s) $g'' \in (G \setminus (I[p + d] \cup J[p]))$. In this case, L_{IJ} will be generated by another overlap involving two motifs I' and J' (with $I' \neq I$ or $J' \neq J$) whose suffix (resp. prefix) list contains g'' and for which the prefix-suffix condition holds. •

Theorem 4 guarantees that no maximal and non duplicated motif is discarded (or, actually, not even generated) because of the prefix-suffix condition. The next result concerns discarded non maximal motifs, and shows that it suffices that only one motif inherits it. As a consequence, the total number of prefixes and suffixes inherited by ℓ-motifs does not exceed the number of maximal motifs of length $\ell - d$. Moreover, this does not have to be a particular motif, but just

any maximal motif (e.g. the first detected). whose extents cover the one that is discarded.

Theorem 5 *Let $L_{M_1}, L_{M_2}, L_{M'}$ be three extents of ℓ-motifs generated at step $i > 2$ such that $L_{M'} \subseteq L_{M_1}$ and $L_{M'} \subseteq L_{M_2}$ and both M_1 and M_2 are maximal. If (wlog) M_1 inherits M' and M_2 does not, then completeness is preserved.*

Proof We show that if both M_1 and M_2 inherit a non maximal (or a duplicated) motif M', then the resulting extents of $(\ell + d)$-motifs would be the same as if only M_1 did, except for possibly one or more non pseudo or duplicated extents. This, together with the fact that pseudo-motifs can not be maximal, ensures that the inheritance of M' by M_2 would be redundant. Let us suppose that $M_1 \neq M_2$ (otherwise the result is trivial). We assume that both M_1 and M_2 inherit M', and then show that any extension generated *only* thanks to the fact that M_2 has inherited M', is either non maximal or a duplication.

Let p' (resp. s') denote any prefix (resp. suffix) of M' that is inherited by M_1 and M_2. Moreover, we denote with P_2 (resp. S_2) the set P_{M_2} (resp. S_{M_2}) of prefixes (resp. suffixes) of M_2 *without* the inheritance of p' (resp. s'). We assume that $p' \notin P_2$, otherwise the result is trivial. Finally, we denote with P_2' the set $P_2 \cup \{p'\}$ (resp. $S_2' = S_2 \cup \{s'\}$).

Since $L_{M'} \subseteq L_{M_1}$ and $L_{M'} \subseteq L_{M_2}$, it must be that $L_{M_1} \cap L_{M_2}$ contains at least $L_{M'}$ and thus it is not empty. Let x denote a generic text position belonging to $L_{M'}$ (and thus also to L_{M_1} and L_{M_2}). We prove the thesis in several different cases of set inclusions relations between L_{M_1}, L_{M_2}, and $L_{M'}$:

1. $L_{M'} = L_{M_1} \cap L_{M_2}$

1a. $L_{M_i} \setminus L_{M'} \neq \emptyset$ for both $i = 1, 2$.
Let o_1 (resp. o_2) be a generic element in $L_{M_1} \setminus L_{M'}$ (resp. $L_{M_2} \setminus L_{M'}$). The extent of M_1 will give raise in general, according to lines 12-14 (possibly filtered), to several extents of $(\ell + d)$-motifs of the form $L_{M_1 N}$ for possibly several ℓ-motifs N. Let us focus our attention to those that contain any $x \in L_{M'}$ (the others do not concern the goal of this proof). There will be, say, h of them that we will denote with $L_{M_1 N_1}, L_{M_1 N_2}, \ldots, L_{M_1 N_h}$ where $L_{M_1 N_i}$ will contain $X_i \subseteq L_{M'}$ for $i = 1, ..., h$. We have that if each set is generated and contains $x \in X_i$, then for our hypothesis it must be that:

(*i*) $\#N_i \in V_\ell[x + d]$, and
(*ii*) $S_{M_1} \cap P_{N_i} \neq \emptyset$.

Moreover, each $L_{M_1 N_i}$ may or may not contain *also* any position $o_1 \in (L_{M_1} \setminus L_{M'})$.
Let us now consider the contribution of M_2 to extents of $(\ell + d)$-motifs, and in particular the extents that result there *only* because

$s' \in S_2$. Notice that the condition (i) above does not depend from whether x is coming from L_{M_1} or L_{M_2}, and thus it also holds for overlaps involving M_2. Moreover, for all extents we are concerned about, we have that also the equivalent of condition (ii) holds, that is, $S'_2 \cap P_{N_i} \neq \emptyset$ because $s' \in (S'_2 \cap P_{N_i})$. Therefore, we have that $L_{M_2N_i}$ will also contain X_i for $i = 1, ..., h$. No other extent will contain elements of M_2 caused by the inheritance of M'. Moreover, if the extent $L_{M_2N_i}$ is generated only because $s' \in S'_2$, it must be that $S_2 \cap P_{N_i} = \emptyset$. In this case, if $\#N_i \notin V_\ell[o_2]$ for any $o_2 \in L_{M_2} \setminus L_{M'}$, then no position other than $x \in X_i$ belongs to $L_{M_2N_i}$, and thus $L_{M_2N_i} = X_i \subseteq L_{M_1N_i}$ for all $L_{M_2N_i}$ generated only because $s' \in S'_2$. Otherwise, if $\#N_i \in V_\ell[o_2]$, then $L_{M_2N_i}$ would have been generated in any case, also without M_2 inheriting M'. Therefore, all such $L_{M_2N_i}$ extents are non maximal or duplicated.

In a similar way, it can be shown that each extent of the form L_{NM_2} that has been generated only because $p' \in (P'_2 \setminus P_2)$ is included (or equal to) an extent $L_{N_iM_1}$. Hence, M_2 inheriting M' has only led to non maximal or duplicated extents.

1b. $L_{M_2} \setminus L_{M'} \neq \emptyset$, but $L_{M_1} \setminus L_{M'} = \emptyset$ (or vice-versa).
We have that $L_{M_1} = L_{M'}$ is a proper subset of L_{M_2}, and thus M_1 is not maximal, contradicting the hypothesis. Similarly, we cannot have that $L_{M_2} \setminus L_{M'} = \emptyset$ and $L_{M_1} \setminus L_{M'} \neq \emptyset$ because M_2 could not be maximal.

1c. $L_{M_1} = L_{M_2}$.
In this case we have that $L_{M_1} = L_{M_2} = L_{M'}$, that is M' is maximal but it is a duplication of both M_1 and M_2, and moreover also M_1 and M_2 are duplications of each other. Notice that since $M_1 \neq M_2$, it must be that they have been generated by means of different prefixes and/or suffixes. This is just a particular (and simpler) case of $1a$ where we have that for each $i = 1, ..., h$ the extents $L_{M_1N_i}$ and $L_{M_2N_i}$ only contains X_i because no o_1 nor o_2 exist, and thus all entries $L_{M_2N_i}$ generated only because $s' \in S'_2$ result in duplications of entries $L_{M_1N_i}$. Similarly, all extents of the form $L_{N_iM_2}$ that have been generated are duplications of extents of the form $L_{N_iM_1}$. This actually holds for any extent that M_2 could generate and in fact in this case also M_2 should have been inherited by M_1.

2. If $L_{M'} \subsetneq L_{M_1} \cap L_{M_2}$

Let x, o_1, o_2 be as above, and let $y \in (L_{M_1} \cap L_{M_2}) \setminus L_{M'}$. Let us consider the same sub-cases a, b and c as in case 1.

2a. $L_{M_1} \setminus L_{M_2} \neq \emptyset$ and $L_{M_2} \setminus L_{M_1} \neq \emptyset$.
This is the most general case and it differs from case $1a$ only in that

the extents $L_{M_1 N_i}$ may now also contain one or more $y \in (L_{M_1} \cap L_{M_2}) \setminus L_{M'}$ next to X_i, or possibly only such y's. But in this case the corresponding $L_{M_2 N_i}$ would contain the very same y's and thus the inclusions still hold.

2b. $L_{M_2} \setminus L_{M_1} \neq \emptyset$, but $L_{M_1} \setminus L_{M_2} = \emptyset$ (or vice-versa).
Similarly to the case 1*b*, this case is again impossible because we have that $L_{M'} \subsetneq L_{M_1} \subsetneq L_{M_2}$ (resp. $L_{M'} \subsetneq L_{M_2} \subsetneq L_{M_1}$) and thus M_1 (resp. M_2) could not be maximal.

2c. $L_{M_1} = L_{M_2}$.
In this case we have that M_1 and M_2 are maximal duplications and that M' is not maximal. This is a simple extension of case 1*c* where nothing else than x_i's and y's would appear in the extent generated by M_1 and M_2, and the latter are equal to the former for the reasons mentioned above.

The arguments above hold for any generic prefix or suffix (inherited or not), and thus they can be extended to the case of sets of them. As a consequence, the result can be iterated and applied to the case of inheritance of inherited lists of prefixes and suffixes, until only maximal and non duplicated extensions are left. •

Finally, observe that a recursive application of Theorem 5 shows that if the list of extensions of a non maximal motif is included into those of $p > 2$ (not necessarily all maximal) ones, then even in this case it is enough that one of them inherits it. Summing up, we have thus proved the following result.

Corollary 1 *At all steps $i > 2$, let $p + 1$ extents $L_{M_1}, L_{M_2}, ..., L_{M_p}$ and $L_{M'}$ of ℓ-motifs be generated such that $L_{M'} \subseteq L_{M_i}$ for $i = 1, 2, ..., p$. If only one of the p motifs inherits M', then the resulting set of maximal $(\ell + d)$-motifs is the same as if all of them (or a part of them) inherit M'.* •

5.4 Complexity

The algorithm consists of $O(k)$ steps. The cost of step i depends from the amount of motifs that have to be overlapped (at most as many as the maximal ℓ_i-motifs) and above all from how many extents they generate before the elimination of non-maximal motifs takes place. We now show a crucial property ensuring that no pseudo-motif is generated at any step.

Theorem 6 *Let IJ be a $(\ell + d)$-long pseudo-motif that could be obtained by overlapping two maximal ℓ-motifs I and J. Then we have that $S_I \cap P_J = \emptyset$.*

Proof Given that IJ is a pseudo-motif with pseudo-extents L_{IJ}, it must be that there exists $d + 1 \leq p \leq \ell$ such that $IJ[p] = I[p] \cap J[p - d]$ and $I[p] \neq J[p - d]$. Moreover, the (pseudo-)motif IIJ obtained by IJ setting $IJ[p] = I[p]$ (resp IJJ setting $IJ[p] = J[p - d]$) has strictly more occurrences that IJ itself. Let

$L_{IIJ} = L_{IJ} \cup Y_1$ (resp. $L_{IJJ} = L_{IJ} \cup Y_2$) with $L_{IJ} \cap Y_1 = L_{IJ} \cap Y_2 = \emptyset$. As a consequence, it must be that $L_I = L_{IJ} \cup V$ and $L_J = (L_{IJ} + d) \cup U$ with $U \subseteq (Y_2 + d)$ and $V \subseteq Y_1$ and thus that both U and V are not empty, $V \cap L_{IJ} = \emptyset$ and $(U - d) \cap L_{IJ}$. Moreover, there is no intersection between V and $U - d$ (otherwise this would have end up in L_{IJ} as well). Given that $V \subseteq L_I$, then any $\ell - d$ long suffix of I must have at least occurred in $V + d$. Also, the fact that $U \subseteq L_J$ means that any prefix of J must have occurred at least in U. The fact that $(V + d) \cap U = \emptyset$ excludes that any $(\ell - d)$-motif could at the same time be (or have been inherited) a suffix of I and a prefix of J. •

Hence, no pseudo-motifs are generated at lines 12-14 due to the prefix-suffix condition and Theorem 6. As a consequence, at each step it is enough to store the extents of all generated motifs, and later on only those of all maximal motifs with their list of prefixes and suffixes (which are at worst as many, given that each eliminated motif leaves a constant size prefix and suffix information). Therefore, the space complexity is the size of the former for which Proposition 4 gave us an upper bound of $O(n \cdot g^k)$. Time complexity in the worst case coincides with the cost of the overlapping phase. The *repeat* starting at line 10 is done $O(k/d) = O(k)$ times and its dominant parts are the nested *for* cycles of lines $12-14$ and the inclusions detection of line 15. The former take $O(n \cdot g^k)$ because this is the maximum number of motifs it generates, and the latter takes $O(n \cdot g^{2k})$ assuming it is made like in [15]. Therefore, overall the time complexity is in $O(k(ng^k + ng^{2k})) = O(kng^{2k})$. Notice that it is linear in the input size.

6 Inferring Relational Motifs

We now show how the algorithm of Section 5 can be extended in order to take into account also relations and how its correctness and completeness is preserved. We name this new algorithm *KMRoverlapR*. As we have anticipated, the novelty starts with the fact that the input sequence is enriched with relations that hold between pair of distinct positions. The inference phase will use this information in order to ensure that relations are conserved as well. Indeed, we still manage the inference storing extents only. These now represent not just repeated motifs, but rather repeated relational motifs. The overlap of two relational *submotifs* of length ℓ that occur at distance d at least q times and in the same relative order necessarily results into a $(\ell + d)$-motif (as before) that also has conserved all relations between pairs of position that are at distance at most ℓ and that come from the same submotif. The only new relations that need to be checked are those between pairs of positions that belong to the two different d-long non overlapped ends of the new motif. In other words, when two ℓ-motifs I and J are overlapped, the relations whose repetitions have to be checked are those between a symbol at the i^{th} position of I for all $1 \leq i \leq d$ and a symbol at the j^{th} position of J for all $\ell - d + 1 \leq j \leq \ell$ for a total of $O(d^2)$ checks to be done. Hence, at each step i, whenever the vector V_i is checked (at lines 6 and 13 that is, when the occurrences of two ℓ_i-patterns are

detected at distance d so that their overlap is a $(\ell_i + d)$-pattern), these $O(d^2)$ comparisons are also made. In order to do so, we use a $n \times d$ matrix W_i that stores in $W_i[j, q]$ the relation groups CR's whose relations hold between position j and position $j + \ell_i - q$ of the input sequence. This is the only kind of relations to be checked at step i. There are two possible results of these $O(d^2)$ checks for the relations. In a first case we can have that in at least q occurrences of the motif the relations are conserved as well, and then a new extent is created per each distinct conserved relation group (this is the case of the 4-motif with extent $\{1, 5, 7\}$ in Example 2). In a second case, no relation is conserved at least q times and the motif is discarded (like the 4-motif with extent $\{2, 6\}$ in Example 2). For all other features of the algorithm, everything can be left unchanged.

It remains to show that the prefix-suffix condition on relational motifs keeps on ensuring that all and only relational pseudo-motifs are discarded. This is stated in the following result.

Theorem 7 *The following results hold:*

1. *Let q be the quorum and let I and J be maximal relational ℓ-motifs such that $|L_{IJ}| = |L_I \cap (L_J - d)| \geq q$ and $S_I \cap P_J = \emptyset$. Then we have that L_{IJ} is either a relational pseudo-extent or a duplication.*

2. *Let $L_{M_1}, L_{M_2}, L_{M'}$ be three extents of relational ℓ-motifs generated at step $i > 2$ such that $L_{M'} \subseteq L_{M_1}$ and $L_{M'} \subseteq L_{M_2}$ and both M_1 and M_2 are maximal. If (wlog) M_1 inherits M' and M_2 does not, then completeness is preserved.*

3. *At all steps $i > 2$, let $p+1$ extents $L_{M_1}, L_{M_2}, ..., L_{M_p}$ and $L_{M'}$ of relational ℓ-motifs be generated such that $L_{M'} \subseteq L_{M_i}$ for $i = 1, 2, ..., p$. If only one of the p relational motifs inherits M', then the resulting set of maximal relational $(\ell + d)$-motifs is the same as if all of them (or a part of them) inherit M'.*

4. *Let IJ be a $(\ell + d)$-long relational pseudo-motif that could be obtained by overlapping two maximal relational ℓ-motifs I and J. Then we have that $S_I \cap P_J = \emptyset$.*

Proof The proofs of parts 1, 2, and 4, are straightforward extensions of those of Theorems 4, 5, and 6 respectively. The proof of part 3 is the extension to relational motifs of Corollary 1, that is a consequence of 1. and 2.. •

Summing up, the correctness and completeness of the algorithm introduced in this section, are based on the following result.

Proposition 5 *A k-pattern is a relational k-motif if, for any $1 \leq d < k/2$:*
(i) Its $(k - d)$-long prefix I is a relational motif (from Theorem 3).
(ii) Its $(k - d)$-long suffix J is a relational motif (from Theorem 3).
(iii) Its relations between all d^2 pairs of positions (l, r) with $1 \leq l \leq d$ and

$(k-d+1) \leq r \leq k$ *are conserved in at least* q *of its occurrences.*
(iv) *It satisfies the quorum (by definition).*
(v) $S_I \cap P_J \neq \emptyset$ *(from Theorem 7).*

where condition (iii) is ensured by the way we have extended the algorithm to the case of relational motifs as described earlier in this section.

From part 4 of Theorem 7, we know that the prefix-suffix condition does not allow to generate any relational pseudo-motif. Hence, we still have that the total size of all the extents of motifs that can be generated at each step is at most as much as the upper bound given in Theorem 2. Hence, the complexity of *KMRoverlapR* changes with respect to that of Section 5 because now also the degeneracy g_R of the relations has to be taken into account. Assuming that d is a constant and thus the cost of the $O(d^2)$ relations tests is negligible, the time complexity of *KMRoverlapR* can be computed as in Section 5.4, except that here the upper bound on the number of maximal motifs of fixed length is that given by Theorem 2. Therefore the time complexity of *KMRoverlapR* is in $O(kn(g^{2k}g_R^{k^2}))$. Reminding that in *KMRoverlapR* the input size is no longer n but rather $n \cdot k$, the complexity is again linear in the input size.

7 Applications to 3D protein structures

7.1 Implementation details

In this section we sketch our implementation of *KMRoverlapR* describing the data structures we use and the way we realize the main steps of the method.

We now list the data structure we keep at each step, assuming that at step i we begin with all and only maximal motifs of length ℓ_i.

- **v** : It is a vector of stacks having the size of input string. For each $1 \leq p \leq n$, the stack at $v[p]$ contains the (identifier of the) maximal ℓ_i-motifs that occur at position p of the input sequence.
- **w** : It is a matrix of size $u \times d$. At each step the entry $w[p, j]$ (with $1 \leq p \leq u$ and $1 \leq j \leq d$) contains the list of relations groups between positions p and j.
- **p** : It is another vector of stacks that contains the same information as **v**, but it is indexed with the motifs' identifiers. It is as long as the number of distinct maximal ℓ_i-motifs, and $p[I]$ contains L_I in a stack.
- **Qa** : It is a vector of stacks of the same size as p and indexed in the same way. The entry $Qa[I]$, indicated in Figure 1 as $\_I$, contains *a list* of extents, each one being the list of occurrences of an $(\ell + d)$-motif obtained with an overlap involving I as a suffix (and a J as prefix which is shown at the beginning of each extent in light grey in Figure 1).

- **Qb** : It is yet another vector of stacks built from Qa by a further filter or subdivision of extents. Each extent contained in $Qa[I]$ is either eliminated if it does not satisfy the quorum, or it is subdivided into one or more new extent(s) each corresponding to the occurrences of a motif with conserved relations as well (a relational motif).

- The list of prefixes and suffixes associated with each motif are stored in a binary search tree.

Figure 1 shows with a simple example the data structures we just described as well as the main steps of the algorithm that we describe as follows.

1. **Init()** : It is executed only once at the beginning and it performs the initialization constructing the first vector v for patterns of length 1. That is, it tells — for each position of the string — which groups contain the letter occurring at that position. In Figure 1 we can see that the identifiers $1, 2$ and 3 are associated to the three groups (motifs of size one coincide with the groups), and that a vector v of size 10 is created indicating where these occur. Moreover, there are three groups of relations that we indicate in black, grey and dashed.

2. **BuildQa()** : It builds the stack vector Qa. We are building motifs of length $\ell_i + d$ from the ℓ-motifs indexed in p. For each stack of p referring to motif I, a list of stacks is built, each one containing the extents $L_{JI} = L_J \cap (L_I - d)$ for an ℓ-motif J such that $L_{JI} \neq \emptyset$ and $S_J \cap P_I \neq \emptyset$. In Figure 1 the list of prefixes and suffixes are not shown at the step describing $BuildQa$ because for $\ell = 1$ they still do not exist. As an example in figure 1, the first position in p is 10 (as in stacks we start from the right) and is the last one in vector v. It is therefore discarded. The second position in p, 9, in pushed in stack "$_1$" of Qa as identifier 1 is v at position 9+1.As we are in stack 1 of p, identifier 1 is also pushed in Qa (in grey, first number in Qa).

3. **BuildQb()** : It builds the stack vector Qb. Each extent of Qa is pushed in Qb in stacks each one corresponding to a newly inferred relational motif. In Figure 1, only needed relations are shown, that is relations between positions j and $j + \ell_{i+1} - 1$ (vector w). For example, the first position in Qa is 4 (starting from the right). There are two relations groups between positions 4 and 5: as we can see in vector w, we have both black and grey relations groups ($CR1$ and $CR2$). Position 4 is therefore pushed in two stacks of Qb (stacks $CR1$ and $CR2$). The grey numbers before position number 4 in Qb (1 and 3) are the prefixes and suffixes for the next step.

4. **BuildV()** : The new vector v is built. Each stack L of Qb is associated to a new identifier corresponding to a relational ℓ_{i+1}-motif and this identifier is pushed in $v[i]$ for each position i occurring in the stack L. For each one of this new identifier, the suffix list is initialized with the number of the entry of Qb where L has been taken (corresponding indeed to the ℓ_i-motif

that was the suffix in the overlap). Similarly, the prefix list is initialized with the number that we have indicated in light grey in Qa of Figure 1 at the beginning of each L (corresponding to the prefix J involved in the overlap with I). At this point the quorum is checked and the motifs that do not satisfy it are discarded and not inserted in v (actually the assignment of new identifiers is made only after this deletions). This is the case in Figure 1 of the extent $\{1\}$ that is crossed in Qa and the extent $\{9\}$ that is crossed in Qb. Now the rightmost entry of v is definitely empty as there cannot be a motif whose occurrences ends after the last position of the sequence, and hence the size of v can be decreased by 1 at each step. From v, its dual vector p is also built showing at position $p[I]$ the extent L_I.

5. **FilterV()** : The vector v is filtered. Here non maximal and duplicated motifs are eliminated. They are detected by means of inclusion tests[2] in the extents listed in the vector p generated by $BuildV()$. As an example, in Figure 1 the entries number 2 and 3 in p are eliminated as shown and their prefixes and suffixes are inherited by the first motif. After this filtering phase, the motifs are re-numbered and a new version of v is generated. The vector v and its lists of prefixes and suffixes are now ready to be the starting point of step $i+1$ of the inference that starts with all distinct maximal motifs of length ℓ_{i+1}.

7.2 Finding repeated substructures: preliminaries

A promising area of application of *KMRoverlapR* is the search for repeated motifs in mono or multidimensional signals or series. In this case we have in general a sequence of points in a multidimensional space. There are then basically two options to represent relative positions in terms of relations. A first way is to define a relation between two points x_p and x_q by discretizing the coordinates of the vector $\overrightarrow{x_q - x_p}$. In the simple case of linear space in which the signal is a sequence of numbers, we can set for example $G_R = \{\{\geq\}, \{<\}\}$, i.e we have either $x_q \geq x_p$ or $x_q < x_p$. For instance, let us consider the input sequence of integers $t =$ 1 4 **8 9 5** 4 8 **7 14 2** 9 4 4, and suppose that we consider only relations, i.e. all values belong to a unique group and hence $g = 1$. Then the relational 3-motif $p1p2p3$ such that $\{p2 \geq p1, p3 < p2, p3 < p1\}$ has the occurrence 8 9 5 at position 3 of t, and the occurrence 7 14 2 at position 8 (they are indicated in bold above). Note that here we have also $g_R = 1$, and hence such relational k-motifs can be extracted in $O(n \cdot k)$ time and space. A second possibility, which is what we actually adopt, is to define the relation between two points x_p and x_q by discretizing the Euclidian distance $d(x_q, x_p)$. In this case, a relational value represents a distance between two points, and thus —

[2]The way this inclusion test is performed deserved a special investigation that we have described in [14].

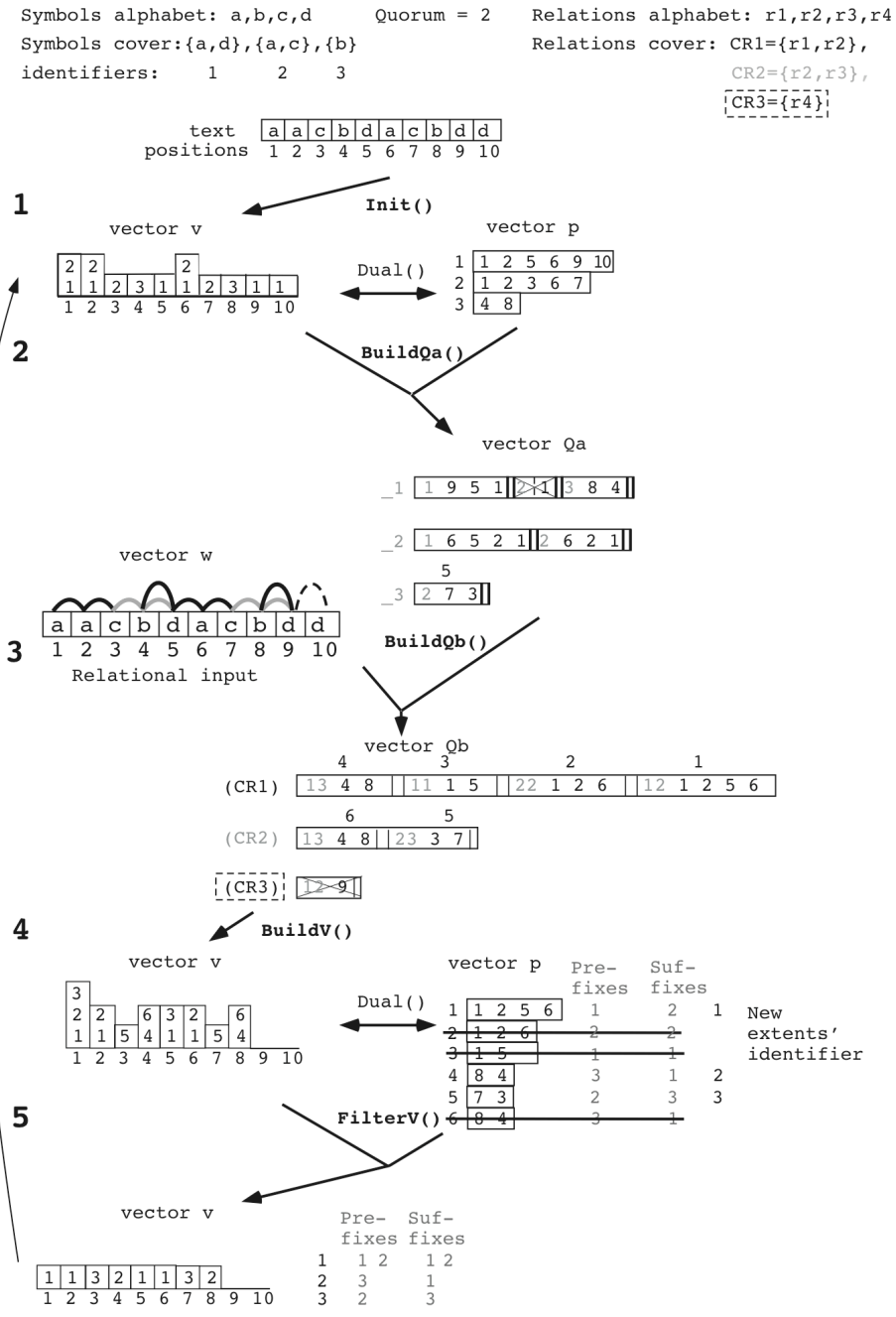


Figure 1: Example with the input string "aacbdacbdd" on $\Sigma = \{a, b, c\}$, the cover $G = \{C_1 = \{a, d\}, C_2 = \{a, c\}, C_3 = \{b\}\}$, and $q = 2$. We omit data structures and procedures (resp. *Qb* and *BuildQb*) concerning relations as they are logically the same as what we do show.

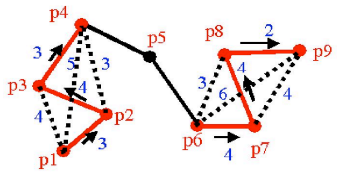


Figure 2: A sequence of 3D points with the distances represented as labels of the edges relating the nodes associated to the points. The two occurrences are $p1-p2-p3-p4$ and $p6-p7-p8-p9$ in a 9-long sequence t.

unlike in the previous case — the motifs' occurrences are insensitive to translations and rotations. Such internal distances between atoms were first used in [3] to search structural motifs in the general context where all the atoms of the protein were considered. In [5], we have a first example in which a degree of approximation is also allowed. As an example of relational motifs representing a structure by using discretized distances as relations, Figure 2 describes two occurrences of a motif of length 4 where we consider that a prior discretization of the distances has been performed so that relations are then positive integers. We then consider a set of relational groups $\{R_j = \{j, ..., j+\delta\}\}$ where δ represents the approximation level: two discretized distances $d(x_p, x_q)$ and $d(x_{p'}, x_{q'})$ belong to the same group whenever $|d(x_p, x_q) - d(x_{p'}, x_{q'})| \leq \delta$. Note that as a consequence we have $g_R = \delta + 1$. In the example of Figure 2, we consider $\delta = 1$ and so $g_R = 2$, and we find two occurrences of a 4-long relational motifs (with $q = 2$).

Hereunder we give an example of the results obtained when searching a structural pattern repeated in the backbone of several proteins. In particular, we report here the results concerning the amount of pseudo-motifs and non maximal motifs, as well as maximal ones, that we have in a series of overlap steps. This data motivates most of the technical results of this paper (we shall see biological results on the same data in Section 7.3). Note that here we have to slightly adapt the algorithm in order to find patterns that occur at least q times in a set of m protein structures. Here $m = q = 4$. The distances between α-carbon are discretized using 1.5Å-long intervals, with a tolerance level $\delta = 2$ and so $g_R = 3$. Here ℓ long motifs are obtained from $(\ell - 3)$-long one, and hence $d = 3$. The average length of the sequences considered is about 400. For each step building maximal ℓ-long motifs overlapping two $(\ell - 3)$-long motifs, we give below the total number of occurrences for the generated ℓ-long motifs, the number of occurrences of pseudo-motifs that are avoided at that step thanks to the prefix-suffix condition, the number of ℓ-motifs that satisfy the quorum $q = 4$, the number of resulting maximal ℓ-long motifs that survive the inclusion test (and that are hence used at the next step), and finally the total number of

occurrences of these latter[3]:

$\ell - 3 \rightarrow \ell$	generated occurrences	avoided occurrences	number of motifs	maximal motifs	occ. of max. motifs
$4 \rightarrow 7$	8,335	0	153	76	16,918
$7 \rightarrow 10$	104,006	77,882	7,565	1,439	84,390
$10 \rightarrow 13$	1,410,871	4,223,925	78,847	4,181	157,311
$13 \rightarrow 16$	4,438,142	23,143,653	97,401	628	4,896
$16 \rightarrow 19$	13,160	31,763	629	25	129
$19 \rightarrow 22$	41	43	5	2	9

It is clear that the amount of occurrences of the pseudo-motifs is considerable. Should they be generated, their detection and elimination would be postponed to the inclusion test, and hence become unbearable. Moreover, these results also show that the number of maximal motifs, and also that of their occurrences, is sensibly smaller than that of generated motifs. This motivates the validity of our choice to focus our attention on maximal motifs. Indeed, although in theory the upper bound on the number of maximal motifs in the worst case is the same as motifs (that is, they can all be maximal), in practice the difference is sensible.

7.3 Some results on cytochromes P450 proteins

Hereunder we give an example of the results obtained when searching a structural pattern repeated in the backbone of several proteins. As anticipated earlier, both the framework of relational motifs and the *KMRoverlapR* solutions we suggested are very general. Depending from the specific application they are used for, they can be instantiated to increase specificity or sensitivity, or to speed up the inference. Observe, for example, that in a 3D protein structure the distances between two positions that are pairwise adjacent are not independent between each other. In other words, consider two positions p and q of the input sequence. Assume that they are part of occurrences of two relational motifs (respectively P and Q) that are distinct but that overlap. Since P (resp. Q) is a relational motif, we know that the distance between positions, say, p and $p+1$ (resp. q and $q+1$) are conserved in the occurrences of P (resp. Q). The same holds for p and $p-1$ (resp. q and $q-1$). When we overlap P and Q, if p and q belong to parts that were not overlapping, the conservation of the relation between this two positions has to be checked. Nevertheless, if the relations between $p-1$ and $q-1$ are conserved as well as those between $p+1$ and $q+1$, then it is unlikely (actually impossible under reasonable degeneracy conditions) that the relations between position p and position q are not conserved. Due to our choice of relations (see below for details), in our applications on 3D proteins we can safely check only relations every, say, 3 positions without affecting the sensitivity of the method, while considerably speeding up the inference. Indeed,

[3]In this experiment only one relation is checked to form a new motif rather than d^2 (see section 7.3 for details).

in the experiment we describe in this section, we have performed overlap steps with $d = 3$ and at each step when we overlap two $\ell - 3$-motifs in order to build an ℓ-motif, we have checked only the relations between the two extreme sides (position 1 and position ℓ) of the new ℓ-motifs. In this way, at each step we only check the conservation of one relation instead of checking it for the $3^2 = 9$ new relations. We have verified that by doing this, we do not miss any biologically meaningful motif. In general, we have that starting from the length ℓ_0, we incrementally infer motifs of length $\ell_0 + d$, $\ell_0 + 2d$, and so on, until length k is reached. If we replace the d^2 relations check with only one between the extreme positions, then the relations that are not tested in a k-motif are those of the set $E(1, k, d)$ where $E(p, k, d)$ for $1 \leq p \leq k$ is recursively defined as follows with $d < \ell_0 < \ell \leq k$.

- $E(p, \ell_0, d) = \emptyset$.
- $E(p, \ell, d) = E(p, \ell - d, d) \cup E(p + d, \ell - d, d) \cup \{(i, j) \mid p \leq i \leq p + d - 1, p + \ell - d \leq j \leq p + \ell - 1, (i, j) \neq (p, p + \ell - 1)\}$.

We chose to study structures of the cytochrome P450 multigenic superfamily (CYP, P450). They are proteins involved in many oxidations of hydrophobic substrates. The substrates are steroid hormones, extracellular fatty acids signalling molecules and vitamins but also exogenous substrates as drugs or environmental pollutants (see [6] for an historical review). These P450s can be found in many living beings: bacteria, yeast, fungi, plants, insects, fishes and mammals. They have been widely investigated notably because of their role in drugs degradation. Their amino-acids primary sequences are dissimilar in spite of their structural similarities. We selected four cytochrome P450 structures: three from bacteria (PDB codes 1CPT, 2HPD chain B and 3CPP) and one from fungi (PDB code 1ROM). Note that here the algorithm searches for patterns that occur at least q times in a set of m protein structures. Here $m = q = 4$. As above, the distances between α-carbon are discretized using 1.5Å-long intervals, with a tolerance level $\delta = 2$ (so that the degeneracy is 3). The average length of the sequences considered is about 400 amino acids.

We focused on the structural motifs found on all of the 4 proteins. The results are the five motifs that are shown in Figure 3. Such motifs were previously identified in [7], using different techniques, on 3 out of these 4 proteins, showing the sensibility of the method we introduced in this paper. In Figure 4 we exhibit the alignments of the structures belonging to different protein sequences. These show that indeed our method succeeds in finding conserved structures, and hence that the relations are a good choice to represent structural information.

8 Conclusions

The first use of relational motifs (without a degenerate alphabet) can be found in [2] to extract repeated RNA secondary structures. In this work an RNA secondary structure was defined as a sequence of *helices*, i.e. as an ordered set

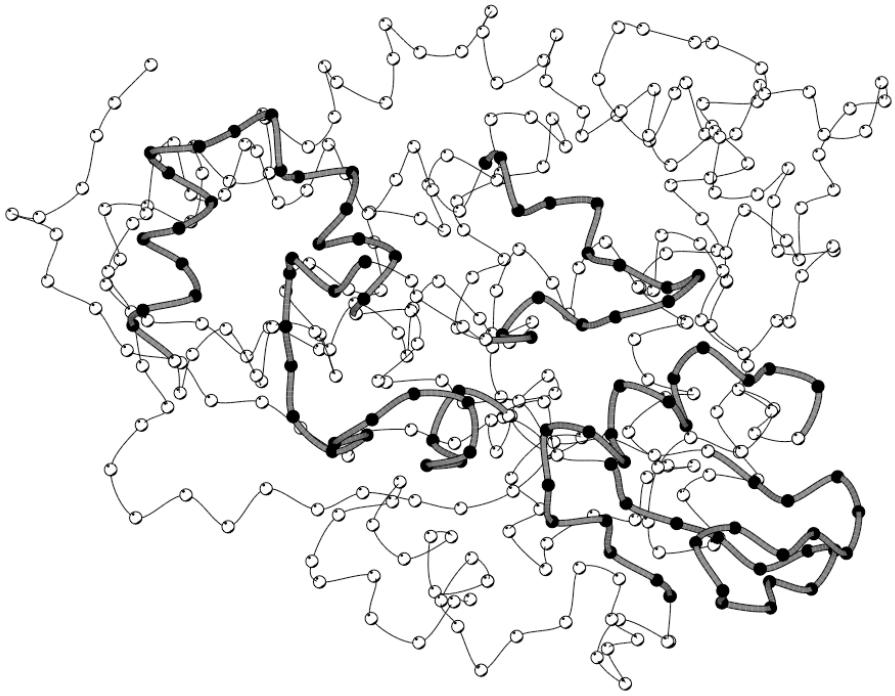
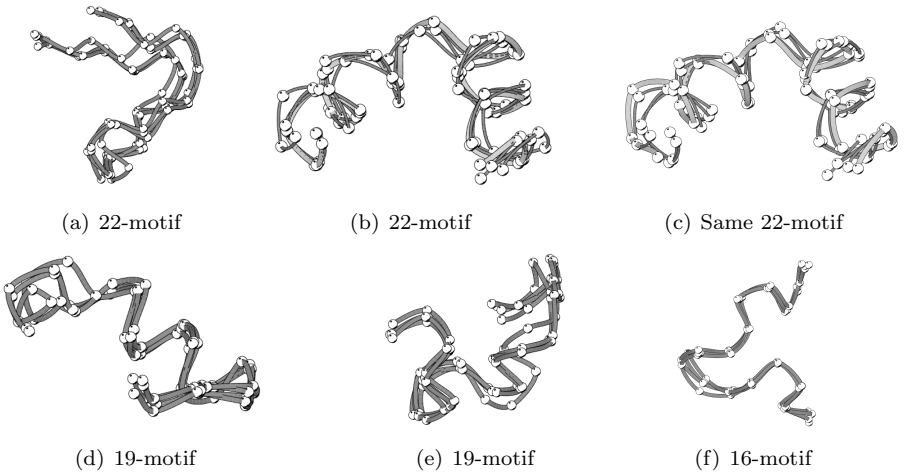


Figure 3: Five structural relational motifs found in four cytochromes P450: two of 22 amino acids, two of 19 and one of 16. They are shown only on protein 3CPP.

of possibly overlapping subsequences of the RNA sequence. In this case the structure is represented as the set of relations (amongst *include*, *overlap* and *disjoint*) between the helices. In [2] an *ad hoc* coding scheme was used allowing to reduce the extraction of such k-long motifs to the extraction of prefixes of k-long words in a dictionary. The overall complexity of the resulting KMR-like algorithm was $O(n \cdot k)$ like *KMRoverlapR* in the case $g = g_R = 1$. However, note that the coding scheme only applies to relations encountered when motifs are union of possibly overlapping intervals [16] as it is the case when dealing with temporal motifs [1]. As a matter of fact, it would be interesting to use the relational variant of *KMRoverlapR* to extract more flexible repeated RNA secondary structures in RNA sequences using a degenerate alphabet to describe the helices, and still using crisp relations ($g_R = 1$) to describe the relations between helices.

Notice that there are applications where actually the number of relations to be taken into account, in the non degenerate case, can be bounded by a constant number that depends from the specific problem addressed. This is the



(a) 22-motif (b) 22-motif (c) Same 22-motif

(d) 19-motif (e) 19-motif (f) 16-motif

Figure 4: Details of the 5 motifs found in the 4 P450. The second one is shown twice (4(b) and 4(c)) as it has two occurrences in protein 1ROM which overlaps (the first one begins at residue 248 and the second one at residue 258, darker on the figures).

case in particular when searching for geometrical motifs in a d-dimensional Euclidian space, like the 3D space used to represent the structure of proteins. In the degenerate case, the interesting one for practical applications, this allows to obtain a good approximation when checking occurrences of a relational k-motif, by only checking a number of relations linear in k and to perform the computation with a fixed length overlap. As a result only $\log k$ steps are performed, the number of k-long motifs is in $O(ng_R^k)$ rather than in $O(ng_R^{k^2})$, and the overall complexity of the relational algorithm is similar to the complexity of *KMRC*.

In Section 7.3 we have shown motifs of different length, namely $16, 19$, and 22 as we have used $d = 3$. Obviously, a 22-motifs contains for example 3 distinct motifs of length 19 that we have discarded because they are certainly less significant than their extension in length since they occur nowhere else. A future work we are planning is to conceive a notion of maximality relating motifs of different length, and possibly an efficient way to infer them.

References

[1] K. Bouandas and A. Osmani. Optimal algorithm for temporal patterns discovery. In *FLAIRS-2003*, pages 455–460. AAAI Press, 2003.

[2] D. Bouthinon and H. Soldano. A new method to predict the consensus secondary structure of a set of unaligned rna sequences. *Bioinformatics*,

15(10):785–798, 1999.

[3] C. W. Crandell and D. H. Smith. Computer-assisted examination of compounds for common three-dimensional substructures. *Journal of Chemical Information and Computer Sciences*, 23(4):186–197, 1983.

[4] N. El-Zant and H. Soldano. Finding repeated flexible relational words in sequences. *Journal of Systemics, Cybernetics and Informatics*, 2(4), 2004.

[5] V. Escalier, J. Pothier, H. Soldano, and A. Viari. Pairwide and multiple identification of three-dimensional common substructures in proteins. *Journal of Computational Biology*, 5(1):41–56, 1998.

[6] R. W. Estabrook. A passion for p450s (rememberances of the early history of research on cytochrome p450). *Drug Metab Dispos*, 31(12):1461–73, 2003.

[7] P. Jean, J. Pothier, P. M. Dansette, D. Mansuy, and A. Viari. Automated multiple analysis of protein structures: application to homology modeling of cytochromes p450. *Proteins*, 28(3):388–404., 1997.

[8] N. C. Jones and P. A. Pevzner. *An Introduction to Bioinformatics Algorithms*. The MIT Press, 2004.

[9] R. Karp, R. Miller, and A. Rosenberg. Rapid identification of repated patterns in strings, trees and arrays. In *Fourth ACM Symposium on Theory of Computing*, pages 125–136, 1972.

[10] M. Lothaire. *Applied Combinatorics on words*. Cambridge University Press, 2005.

[11] L. Marsan and M.-F. Sagot. Algorithms for extracting structured motifs using a suffix tree with application to promoter and regulatory consensus identification. *Journal of Computational Biology*, 7:345–360, 2001.

[12] N. Pisanti, M. Crochemore, R. Grossi, and M.-F. Sagot. A basis of tiling motifs for generating repeated patterns and its complexity for higher quorum. In B.Rovan and P.Vojtás, editors, *Mathematical Foundations of Computer Science*, LNCS 2747, pages 622–631. Springer-Verlag, 2003.

[13] N. Pisanti, H. Soldano, and M. Carpentier. Incremental Inference of Relational Motifs with a Degenerate Alphabet. In *Combinatorial Pattern Matching (CPM)*, pages 229–240. Springer-Verlag, 2005. LNCS 3537.

[14] N. Pisanti, H. Soldano, M. Carpentier, and J. Pothier. Implicit and explicit representation of approximated motifs. Technical Report TR-05-19, Dipartimento di Informatica, Università di Pisa, 2005.

[15] H. Soldano, A. Viari, and M. Champesme. Searching for flexible repeated patterns using a non-transitive similarity relation. *Pattern Recognition Letters*, 16:243–246, 1995.

[16] S. Vialette. On the computational complexity of 2-interval pattern matching problems. *Theoretical Computer Science*, 312(2-3):223–249, 2004.