A Relational Extension of the Notion of Motifs: Application to the Protein Common 3D Substructures Searching Problem

Nadia Pisanti, Henry Soldano, Mathilde Carpentier, Joël Pothier

Journal of Computational Biology: http://mc.manuscriptcentral.com/liebert/jcb

A Relational Extension of the Notion of Motifs : Application to the Protein Common 3D Substructures Searching Problem

Journal:	*Journal of Computational Biology*
Manuscript ID:	JCB-2008-0019.R1
Manuscript Type:	Original Paper
Date Submitted by the Author:	24-Sep-2008
Complete List of Authors:	Pisanti, Nadia; University of Pisa, Computer Science Soldano, Henry; Université Paris 13, LIPN-UMR 7030 CNRS; Université Paris VI, Atelier de BioInformatique Carpentier, Mathilde; université Pierre et Marie Curie-Paris6, Equipe de Génomique Analytique Pothier, Joël; Université Pierre et Marie Curie-Paris6, Atelier de BioInformatique
Keyword:	strings, algorithms, combinatorics, protein motifs, PROTEIN STRUCTURE
Abstract:	Geometrical configurations of atoms in protein structures can be viewed as approximated relations between them. Then, finding similar common substructures within a set of protein structures belongs to 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 \emph{relational motifs}. For this class of problems we give an algorithm that is a suitable extension of the KMR paradigm and, in particular, of the KMRC as it uses a degenerate alphabet. The algorithm contains several improvements with respect to KMRC that become especially useful when---as it is required for relational motifs---the inference is made by partially overlapping shorter motifs, rather than concatenating them. The efficiency, correctness and completeness of the algorithm is ensured by several non-trivial properties that we prove in this paper. Finally, we give some examples in the important field of protein common 3D substructures searching. The methods implemented have been tested on several examples of protein families: serine proteases, globins and cytochromes P450. The detected motifs have been compared to those found by multiple structural alignments methods.

A Relational Extension of the Notion of Motifs : Application to the Protein Common 3D Substructures Searching Problem

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

‡ Dipartimento di Informatica, Università di Pisa,
Largo B. Pontecorvo, 3 I-56127 Pisa, Italy, pisanti@di.unipi.it

* LIPN - UMR 7030 CNRS - Université Paris 13,
Av. JB-CLément, F-93430 Villetaneuse, France, henry.soldano@lipn.univ-paris13.fr

◇ Université Pierre et Marie Curie-Paris6, Atelier de BioInformatique,
12 rue cuvier, 75005, Paris, France, jompo@abi.snv.jussieu.fr

° Université Pierre et Marie Curie-Paris6, Equipe de Génomique Analytique, INSERM511
91, bd de l'Hôpital 75013 Paris, France, mathilde@abi.snv.jussieu.fr

September 24, 2008

keywords :

Abstract

Geometrical configurations of atoms in protein structures can be viewed as approximated relations between them. Then, finding similar common substructures within a set of protein structures belongs to 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 (Karp *et al.*, 1972) paradigm and, in particular, of the KMRC (Soldano *et al.*, 1995) as it uses a degenerate alphabet. The algorithm contains several improvements with respect to (Soldano *et al.*, 1995) that become 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 (Karp *et al.*, 1972). The efficiency, correctness and completeness of the algorithm is ensured by several non-trivial properties that we prove in this paper. The algorithm have been applied in the important field of protein common 3D substructures searching. The methods implemented have been tested on several examples of protein families: serine proteases, globins and cytochromes P450. The detected motifs have been compared to those found by multiple structural alignments methods.

1 Introduction

Finding repeated subsequences and substructures in biological (resp. sequential and structural) data is having growing importance for various different applications in molecular biology. Among them we can mention the detection of trasncription factors binding sites as repeated gapped motifs in the upstream regions preceeding genes, or the prediction of RNA secondary structures as complementary reversed repeated subsequences, the detection of common fragments of genomic sequences as a starting point of measuring genomic distances, etc. In this paper we focus on yet another biological application, that is the detection of common substructures in 3D proteins. In a preliminary version of this paper (Pisanti *et al.*, 2005) we have designed an algorithm for the inference of repeated motifs under the new framework of *relational* motifs which results particularly suitable for this purpose. The present paper extends this work, presenting all the related theoretical results, and discussing our experiments about the above mentioned repeated structural fragments extraction problem.

Motifs inference in biological applications requires a certain degree of approximation in establishing whether a biological object is basically the same as another one. For this reason, the possibly huge size of solutions in the search space makes the algorithmical solution tricky. It is very difficult to find the right balance between the sensitivity of a motif inference tool and its efficiency when an exhaustive algorithmical approach is suited. Most of the difficulty comes from the unavoidable noise of biological data which causes an explosion of intermediate candidates (typically, shorter motifs to be extended or composed to make longer ones). Hence, it is very important that the inference tool offers a way to refine the query in order to minimize this noise. We define a new class of problems that extends the traditional inference of repeated motifs. The 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 (Lothaire, 2005) or Chapter 4 of (Jones and Pevzner, 2004), or (Parida, 2008). 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 (Marsan and Sagot, 2001) or of defining slim *generators* for the complete set of the motifs (Pisanti *et al.*, 2003). The refinement of the former has clear motivations in molecular biology in inferring transcription fac-

tors 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. In Section 7 we will focus our attention on the application in structural molecular biology, that is finding repeated substructures in 3D protein structures, using as relations the distances between the α-carbons in the protein structure. Notice that this data differs from that used in (Feng *et al.*, 2005; Parida and Zhou, 2005) where the authors propose a combinatorial pattern discovery technique to investigate protein folding trajectory data from simulated experiments: the data used there is not the amino acids sequence of the protein primary structure, nor just the final configuration of the protein, but the various *reaction coordinates* of key intermediate states of the folding process of the simulation. These data are then combined and represented by (normal, that is non relational) patterns that are suitably inferred and clustered, allowing to extract structures that were overlooked in previous work.

The inference of motifs with relations introduced in the present paper 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 or Hoogsteen 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.

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 (Karp *et al.*, 1972), we will have to deal with the degenerate alphabet like in KMRC (Soldano *et al.*, 1995). In particular, we will restrict our attention to *maximal* motifs for a notion of maximality which is the same as in (Soldano *et al.*, 1995). 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 (Karp *et al.*, 1972) in the same direction as (El-Zant and Soldano, 2004) where relations in motifs had been introduced for the first time. In (El-Zant and Soldano, 2004), 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 (El-Zant and Soldano, 2004) by decreasing of an exponential factor the amount of candidates generated at each step that we prove to be redundant. We will present various propositions, lemmas and theorems whose detailed proofs are left in the Appendix (section 9).

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 (Soldano *et al.*, 1995) by extending the method of (Karp *et al.*, 1972) to the case of maximal motifs which are approximate in that they are expressed using the degenerate alphabet. Basically, in (Soldano *et al.*, 1995) like in (Karp *et al.*, 1972), 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 (Soldano *et al.*, 1995) 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 (Soldano *et al.*, 1995) is different from that of (Karp *et al.*, 1972) 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 motifs 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 (Soldano *et al.*, 1995) 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 (Soldano *et al.*, 1995). 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 the application to finding 3D substructures in tertiary structures of a set of proteins. We consider the protein 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 common 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 overlapping 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 effect 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 (Marsan and Sagot, 2001). 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 (Karp *et al.*, 1972). 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.*

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.*

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.*

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 (Soldano *et al.*, 1995), but there it was not so relevant because there was only one overlap step. In the case of a series of overlap steps such as those we have here, the absence of this optimization step could result in catastrophic effect 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):

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

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 large 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 $L = \{1, 2, \ldots, n-k+1\}$, 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, 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 large 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 (Soldano *et al.*, 1995), it can be useful to view the upper bound on the number of k-motifs also in terms of g. In (Soldano *et al.*, 1995) 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.

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.*

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 $k/2 \leq \ell < k$.*

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.

[1] In (Soldano *et al.*, 1995) the result is stated for maximal motifs. However the very same proof works for motifs in general.

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 generating 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 $n(g^\ell.g_R^{\ell(\ell-1)/2})$.*

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 (Soldano *et al.*, 1995), 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 one 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 in this form 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 $\ell + d$.*

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.*

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 $S_I \cap P_J = \emptyset$.*

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 (Soldano *et al.*, 1995). 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 $S_I \cap P_J = \emptyset$.*

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 Searching common substructures in a set of 3D protein structures

A promising area of application of *KMRoverlapR* is the search for repeated motifs in mono or multidimensional signals or series. In the case of protein structures we have in general a sequence of points in a multidimensional space.

We define the relation between two points x_p and x_q by discretizing the Euclidian distance $d(x_q, x_p)$. Then a relational value represents a distance between two points, and thus the motifs' occurrences are insensitive to translations and rotations. This method has been implemented (in C language) in order to search for structural motifs in protein structures. Each backbone C_α is a point in the three dimensional space. The amino acid sequence may be ignored or not: in addition to relations a cover on the amino acid alphabet can be set. For example a cover of the alphabet may be the groups of non polar amino acids (ACFGILMPTVWY), polar amino acids (DEHKNQRSTYC) and small amino acids (ADGNPSTV).

There are many methods to build a multiple alignment of protein structures, but very few are designed to find structural motifs.

Most of multiple alignment methods are built on the same blueprint: i) all pairwise structural alignments are computed, ii) a dendogram is built based on the pairwise alignment scores, iii) all structures are multiply aligned in the order given by the previous tree. The differences between this kind of methods lie in the pairwise alignment algorithm and the dendogram calculation algorithm (mostly UPGMA). Some well known pairwise structural alignment methods have been adapted to multiple structural alignment. For example CE-MC (Guda *et al.*, 2001) is an adaption of CE (Shindyalov and Bourne, 1998), POSA (Ye and Godzik, 2005) of FATCAT (Ye and Godzik, 2003), and MALECON (Ochagavia and Wodak, 2004) of Boutonnet et al. algorithm (Boutonnet *et al.*, 1995).

The methods COMPARER (Sali and Blundell, 1990), MATRAS(Kawabata, 2003) and MAMMOTH (Ortiz *et al.*, 2002; Lupyan *et al.*, 2005) have also been adapted to multiple structural alignment. MUSTANG (Konagurthu *et al.*, 2006) is a recent method using the same procedure of progressive pairwise heuristic. Pairwise structural alignment methods are well described in in several reviews (Brown *et al.*, 1996; Eidhammer *et al.*, 2000; Taylor *et al.*, 2001; Carpentier and Pothier, 2007). Some other methods are using one structure as pivot and align other structures using this pivot as a reference (Gerstein and Altman, 1995; Gerstein and Levitt, 1996, 1998; Wu *et al.*, 1998b,a; Ye and Janardan, 2004; Escalier *et al.*, 1998; Leibowitz *et al.*, 1999, 2001). The algorithm developed by Crandell and Smith (Crandell and Smith, 1983), and implemented by Brint and Willet (Brint and Willett, 1987) use a graph in order to represent protein atoms. This method was the first to align several protein structures by using graph algorithms. Since other methods have been implemented (Koch *et al.*, 1992, 1996; Cook and Holder, 1994; Su *et al.*, 1999; Dror *et al.*, 2003b,a). The method of Koch et al. search for common secondary structures by representing protein with a graph of secondary structures.

Our method is not designed to find a multiple structural alignment. Rather it finds multiple structural motifs. The extents of a motif may contain several occurrences in the same protein. However we have to test the relevance of the found structural motifs. Therefore we compared our motifs to those obtained by some multiple structural alignment programs. The aim is to determine wether we find the evolutionary conserved motifs. We have also studied the features of the motifs found by the method. To test our method we chose to compare the well known globin structures, that are often used when testing multiple structural alignment methods. For example it has been used to test the MUSTANG program (Konagurthu *et al.*, 2006). We chose also to compare our results with those obtained for the serine protease family (also from this article). We also run our algorithm on the structure of cytochromes P450 multigenic superfamily (CYP, P450). These proteins are involved in many oxidations of various and numerous hydrophobic substrates (see Estabrook (2003) for an historical review). lTheir amino-acids primary sequences are dissimilar in spite of their structural similarities. All PDB codes are given in table 1.

First we studied the effects of varying algorithm parameters on structural motif lengths and number. Second we compared motifs obtained for the serine proteases to the manually curated alignments and MUSTANG's alignments. Third we compared the globin structural motifs to MUSTANG and POSA alignments. We chose the latter two families because they are widely studied: serine protease mainly contain *beta* strands and globins are made of α helices. At last we studied more carefully the motifs found for those families. For the P450 family two structure sets were used : one containing all P450s in the PDB cluster 50 set (protein structures sharing less than 50% of amino acids sequence identity) and the other set made of all SCOP (Murzin *et al.*, 1995) domains sharing less than 40% of sequence identity (extracted from ASTRAL (Brenner *et al.*, 2000)

list[2]).

Table 1: Protein structure data sets

Data Set	Number of structures	PDB/SCOP codes and chain
Globins		
Set 1	5	1hho_A 1hho_B 1mbd_A 2dhb_A 2dhb_B
Set 2	9	1eco_A 1hbg_A 1hho_A 1hho_B 1mbd_A 2dhb_A 2dhb_B 2lh7_A 2lhb_A
Set 3	11	1dlw_A 1eco_A 1hho_B 1mbd_A 2dhb_B 4vhb_A 1dly_A 1hho_A 1idr_A 2dhb_A 2lh7_A
Serine proteases		
Set 1	7	1ppb_H 1ton_A 2pka_B 3est_A 3rp2_A 5cha_B 5ptp_A
Set 2	13	1arb_A 1sgt_A 2alp_A 2sga_A 3est_A 3sgb_E 5ptp_A 1ppb_H 1ton_A 2pka_B 2snv_A 3rp2_A 5cha_B
Cytochromes P450		
Set 1	35	1izo_A 1izo_C 1io9_B 1cpt 1q5e_A 1odo_A 1n97_A 1oxa 1ued_B 1akd 1o76_B 1gwi_A 1w0e_A 1egy_A 2rom 1lgf_A 1lfk_A 1ued_A 1n97_B 1n40_A 1n4g_A 1s1f_A 1e9x_A 1h5z_A 1gwi_B 1pkf_A 1suo_A 1dt6_A 1cl6_A 1w0g_A 1jpz_B 1n6b_A 1f4t_A 2bmh_A 1po5_A
Set 2	16	d1cpta_ d1jfba_ d1n97a_ d1q5da_ d1tqna_ d2ij2a1 d1io7a_ d1lfka_ d1odoa_ d1re9a_ d1x8va_ d1izoa_ d1n40a_ d1po5a_ d1s1fa_ d1z8oa1

7.1 Finding repeated 3D substructures: parameters

We represent a structure by using discretized distances as relations. As an example of relational motifs representing a structure, Figure 1 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 1, we consider $\delta = 1$ and so $g_R = 2$, and we find two occurrences of a 4-long relational motifs (with quorum $q = 2$). Notice that here we had slightly adapted the algorithm in order to find patterns that occur at least q times in a set of m protein structures. Here $m = q = 2$.

The parameters are: the mesh for the discretization, the margin δ for the relational groups and the quorum q. As the distance between 2 successive C_α is always around $3.8\mathring{A}$, the smallest motif sizes are 3 residues. As anticipated earlier, both the framework of relational motifs and the *KMRoverlapR* solutions we suggested are very general. Depending on the specific application they are used for, relations can be instantiated to increase specificity or sensitivity, or

[2]ASTRAL SCOP 1.73 genetic domain sequence subsets, based on PDB SEQRES records (i.e. sequences), with less than 40% identity to each other

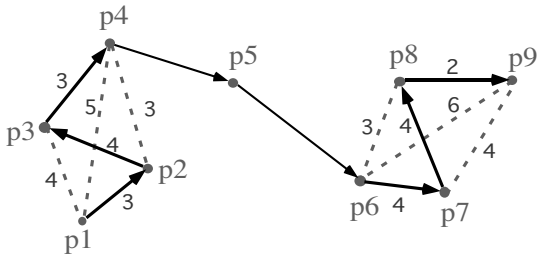

Figure 1: A sequence of 3D points with the distances represented as labels of the edges relating the nodes associated to the points. The two occurrences are $p1 - p2 - p3 - p4$ and $p6 - p7 - p8 - p9$ in a 9-long sequence t.

to speed up the inference. One can observe, for example, that in a 3D protein structure the distances between two sets of adjacent positions are not independent. 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 $q+1$, 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. For example if we perform steps with $d = 3$, at each step, we overlap two $\ell - 3$-motifs in order to build an ℓ-motif and we only have to check the relation between the two extreme sides (positions 1 and ℓ) of the new ℓ-motifs instead of checking the $d^2 = 3^2 = 9$ new relations. We have verified that by doing this, we do not miss any biologically meaningful motif. Usually 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.

We also tried to fix the number of overlapping positions. For example, with an overlapping o, we build motif of length $2(\ell - o)$ from two motifs of length ℓ. With this procedure, motifs length growth is much faster but some motifs, different from maximal length motifs, are shorter than they could be with the classical procedure $d = 1$.

We tried several mesh: $0.5\mathring{A}$, $1\mathring{A}$, $1.5\mathring{A}$. When searching for structural motifs there is always a difficult step (i.e. a motif length) to bypass, because too many motifs are generated (see figure 2). This peak is due to canonical secondary structures (mostly α helices) which have very small structural variations and

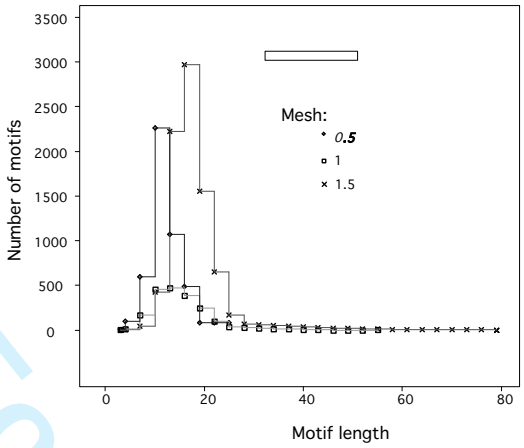

Figure 2: Number of motifs found at each step (length) with several meshes for the structures of globins set 1.

therefore may generate many extents. In most cases motifs are longer with a $1.5\mathring{A}$ mesh but sometimes again too many motifs are generated. This may also occurs with a mesh of $1\mathring{A}$. The best results are obtained for a mesh of $1\mathring{A}$. If the margin is greater than 1, again too many motifs are found. Computing time increases with the number of proteins but rather linearly (see for example figure 3) but it strongly depends on data nature.

7.2 Comparison to structure multiple alignments

Several parameters where tried for all the following sets: for each mesh of $0.5\mathring{A}$, $1\mathring{A}$ or $1.5\mathring{A}$ we tried $d = 1, 2, 3, 4$ or $o = 3, 4, 5$.

7.2.1 Comparison of the motifs to manually curated alignment of serine proteases

The first set of serine proteases is composed of mammalian proteins and the second is made of more divergent serine proteases. We compare the ungapped blocks of Lesk and Fordham's alignments (Lesk and Fordham, 1996) to the motifs found by our method. There were 12 blocks longer than 4 residues in set 1 serine protease alignment and 5 blocks in set 2.

For the set 1, all sets of parameters but one (mesh $1.5\mathring{A}$ and $d = 1$) do not implicate the generation of too many motifs. The results are almost the same with $d = 1, 2, 3, 4$ but if we use the fixed size overlapping procedure ($o = 3, 4 or 5$) , motifs are shorter. We chose to study more precisely the results found with $d = 3$.

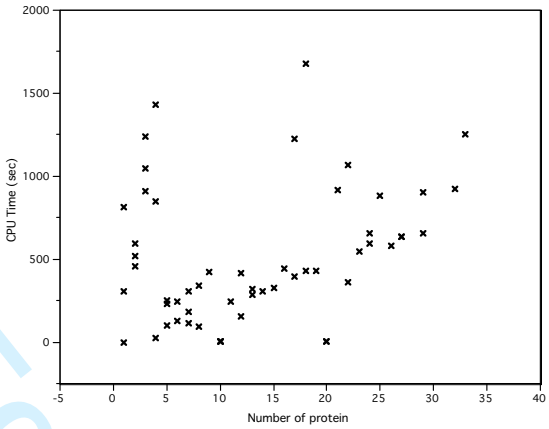

Figure 3: CPU time *vs.* number of proteins for the set 1 of cytochrome P450 (mesh 1, overlapping 3). Times were measured for several subsets of the P450 set 1. Depending on the proteins, times may be very different.

The two first conserved blocks (positions 16 to 25 and 26 to 35 in all PBDs) are found by our method in only one motif. These two blocks are separated by one position in each proteins of the original alignment but the residues are not all aligned at this position. With a 0.5$\mathring{A}$ mesh, we find a block of length 13, shorter than the reference block. But with a 1$\mathring{A}$ mesh, the block length is 18 residues and with 1.5$\mathring{A}$, 21 residues: the whole are found (see figure 4). In MUSTANG alignment, the two blocks are also merged and the block length is 19 residues. All other blocks are found by our method but they may be in several parts (which may overlap) with a 0.5$\mathring{A}$ mesh. For example, the longest one (26 residues, beginning at residue 100) is also the longest motif found by our method (28 or 25 residues depending on the mesh), but with a mesh of 0.5$\mathring{A}$ the longest motifs are 13 residue long (two words of length 13 covering 18 residues of the 26 residues block). It is also interesting to remark that sometimes motifs occur several times in proteins. For example the fourth block (around positions 62-73, depending on the proteins) corresponds to a motif with several occurrences in one protein.

MUSTANG alignment have been compared to the reference Lesk and Fordham's alignment. MUSTANG alignment shows more blocks (205 positions without gap in MUSTANG alignment *vs.* 174 in the manual alignment (Konagurthu *et al.*, 2006)). Some of these positions are at the extremities of reference alignment blocks. Our blocks mostly agree with those new positions. Others are isolated and our method cannot find them. One block is much more longer in MUSTANG alignment but our method found the same boundaries as the reference alignment, and consequently disagrees with MUSTANG.

There are more proteins in the second set, and it not possible to compute

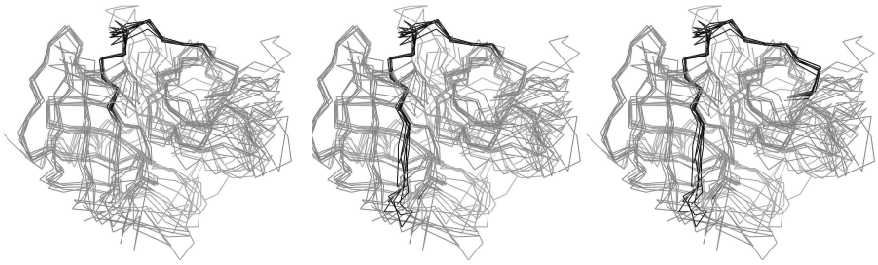

Figure 4: Motifs in serine protease set 1. The motif contains the same positions as the reference alignement. From left to rigth, the mesh is $0.5\mathring{A}$, $1\mathring{A}$ and $1.5\mathring{A}$ (margin 1, shift $d = 3$). As the mesh increases, the motif is longer. All proteins are superimposed according to the longest motif (the same rotation-translation matrices are used for the three pictures).

the motifs with $d = 1$. These proteins are far more divergent. Consequently, there are less blocks in the reference alignment and they are smaller. Similarly, our motifs are smaller. The longest motifs are of length 13. We do not find a motif occurring only once in each protein. It is possible to find the reference blocks among the motifs, but many other occurrences of each motif are found in each protein (see for example figure 5).

In these examples we have seen that all conserved blocks found by alignment methods are found by our method. But many other motifs are also found and it may be interesting to wonder what kind of substructures they are. Many of the motifs are used to build motifs of a greater size but many positions disappear during these steps. However these intermediate motifs are interesting because they show substrctures which occurs several times in one protein. Obviously most of these intermediate motifs are canonical secondary structures (α helices and β strands) but some motifs are different. Maybe these motifs could be used to characterize the protein family.

7.2.2 Comparison to Globin alignments (Konagurthu *et al.*, 2006)

The set 1 of globins contains mammalian proteins, in set 2 invertebrate and plant globins are added to set 1 and in set 3 truncated globins are added to set 2.

The globins are shorter than the proteases but it is more difficult to find relevant motifs because there are many α helices. Usually when many helices are present, best results are obtained with mesh $1\mathring{A}$. If the mesh is smaller or greater, too many motifs may be generated. With $d = 3$ and $mesh = 1\mathring{A}$ we found a 46 residue-long motif which corresponds to the longest blocks of POSA and MUSTANG alignments. It is possible to find motifs everywhere on the molecules except in one variable region which corresponds to a region with

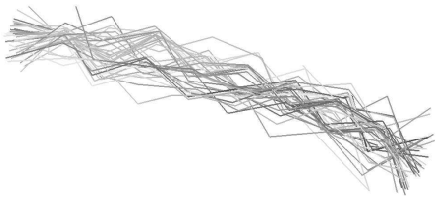

Figure 5: Example of 11 residues motif in serine protease set 2 (mesh 1$\mathring{A}$, shift $d = 3$). There are many occurrences (21 occurrences for 13 proteins) but they all are structurally similar.

gaps in MUSTANG alignment (see figure 6). With proteins of set 2 the longest motifs have 22 residues ($d = 3$, mesh=1), but there are 74 motifs of length 22 in those globins and they all are α helices. The other shorter motifs contains the blocs found by POSA or MUSTANG which always agreed except few positions at the extremities.

As the set 3 contains 3 truncated globins added to the 9 previous proteins, it is interesting to set the quorum $q = 9$ and to check wether our method finds the motifs found in the set 2. These 3 truncated proteins miss the beginning of the globins. Therefore the first block (16 residues in POSA, 17 in MUSTANG) is not present in these 3 proteins. However we do find the first motifs in the other proteins. More motifs are generated with a quorum of 9/11 proteins but with $d = 3$ the maximal motif is also 22 residues.

7.2.3 Sequence constraints

Two sets of P450 have ben studied (see table 1), one with 35 proteins which amino acid sequences share less than 50% identity and one with 16 proteins (sequence identity $< 40\%$). Same sets of parameters were used for the study of cytochromes P450, but this set of 35 structures with many α helices generate too many motifs in all cases. Motifs are found for the 16 proteins set with $mesh = 1\mathring{A}$ and a fixed overlap $o = 3$. This examples are very difficult cases for our method. Therefore we added some restrictions on the nature of the amino acids: they have to be either polar or apolar or small in order to be at the same position in one motif (see page 21 for detailed categories). The number of motifs is then extremely smaller and we can compute motifs with $d = 1$ even for 35 structures. However the longest motifs are shorter than those found in the set 2 of P450s without sequence restriction (length of 8 or 9 instead of 22 residues). These small motifs are mostly fragments of α helices. Thus sequence

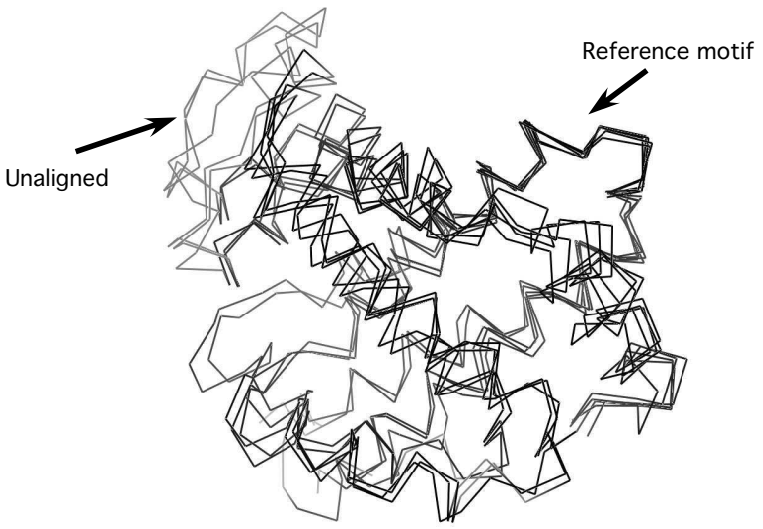

Figure 6: Globins set 1. No motif is found in the upper left grey region. Everywhere else, there are some motifs. Grayscale indicates depth. Proteins are superimposed according to one motif of 15 residues (reference motif).

constraints may be too strong in order to search for structural motifs in such sequence divergent proteins.

We also tried the same cover on the alphabet for the globin sets and motifs are also shorter.

7.2.4 Motifs quality

In globins set 1 we computed RMSDs for all motifs longer than 6 residues. RMSDs are lower than $2.0\mathring{A}$ and increase with motif length until only well conserved motifs are present (see figure 7).

When increasing the shift d, the shorter motifs found show slightly higher RMSD (around $0{,}15\mathring{A}$ in average, see figure 7).

We also computed RMSDs for motifs obtained for several mesh and $d = 3$ and as expected RMSDs are higher with mesh$= 1.5\mathring{A}$.

8 Conclusions

The first use of relational motifs (without a degenerate alphabet) can be found in (Bouthinon and Soldano, 1999) to extract repeated RNA secondary structures. In this work an RNA secondary structure was defined as a sequence

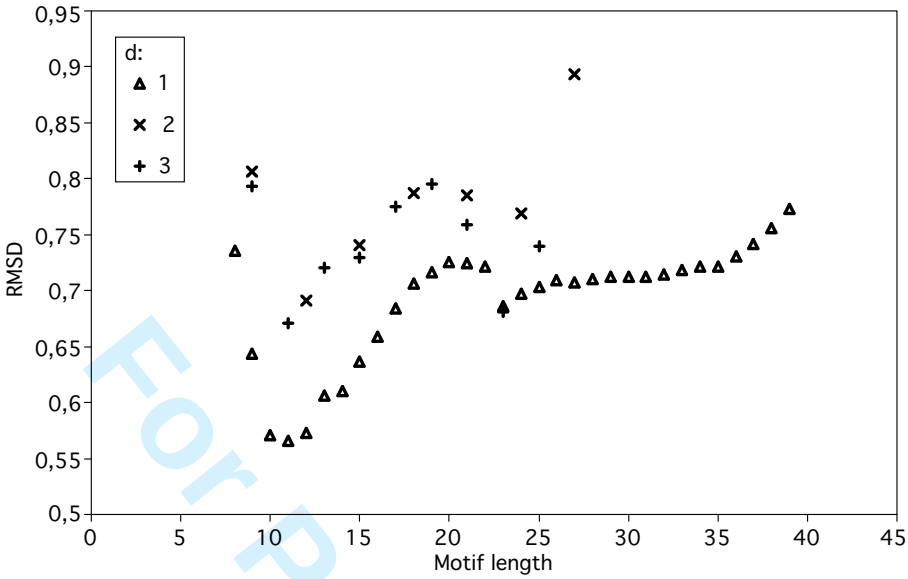

Figure 7: Globins set 1. RMSD average for all motifs of a given length *vs* motif length for shifts $d = 1, 2, 3$. The RMSD of each motifs has been computed for several parameters.

of *helices*, i.e. as an ordered set 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 (Bouthinon and Soldano, 1999) 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 (Vialette, 2004) as it is the case when dealing with temporal motifs (Bouandas and Osmani, 2003). 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 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 de-

generate 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*. We have used a fixed length overlap in the experiments on P450 proteins, thus avoiding motif lengths producing too much computations. However we have found that whenever all maximal lengths motives are searched for, a fixed length overlap may result in missing some relevant motives corresponding to non investigated lengths.

We have investigated and discussed the extraction of structural motifs from a set of protein structures. However the question of their use to search databases is also important. In a previous paper (Pisanti *et al.*, 2006) we have investigated issues related to the representation of our relational motifs and their search on protein structures out of the set from which the motifs have been extracted. One of the main limitation of the method is the number of structures that can be handled when extracting motifs. This is due to the fact that the search is here exhaustive. One way, not experimented so far, to preserve exhaustivity while allowing to consider larger sets of structures is to previously cluster the whole set of proteins structures and then incrementally extract the motifs. A first cluster (of say a dozen of structures as shown here) of proteins will give a set of motifs then such motifs may be considered as standalone structures and added to a second cluster in order to produce new motifs using again our extraction method. Note that new motifs are then necessarily included in previous ones, and so such a procedure should hopefully converge to a small but significant set of motifs representative to the whole family. A second limitation of the method, that can also be a drawback for the above variant, is that in some cases many motifs happen to be extracted. These motifs are however very similar when considering their occurrences on the structures. A way under investigation to overcome this limitation is obviously to define a notion of similarity between motifs relying on their extensions, i.e. their set of occurrences on the investigated structures. This procedure should drastically reduce the number of motives and it would be then possible to build a structural alignment from them.

The method could be of interest for the structural characterization of protein structure families. It could be run on all protein families of SCOP (Murzin *et al.*, 1995) for example, in order to find motifs specific of each families (structural signature).

As the method exhibits similar substructures in a structural family, it could be an help in building structural cores for threading methods (Marin *et al.*, 2002) or in building structural alphabets as those used by Baker and coll. in their *ab initio* folding method (Simons *et al.*, 1999).

9 Appendix

9.1 Proof of proposition 1

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 $O(k^3 n)$. •

9.2 Proof of proposition 2

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 submotifs 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 at most n positions for which the check has to be done (for all the motifs that are being generated).

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. •

9.3 Proof of proposition 3

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 $\sum_{i=1}^{k/d} n \cdot d^2 = ndk \in O(n \cdot k)$. •

9.4 Proof of lemma 1

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. •

9.5 Proof of theorem 1

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 $L_p \cap (L_s + \ell - k)$. •

9.6 Proof of theorem 2

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 $n(g^\ell.g_R^{\ell(\ell-1)/2})$. •

9.7 Proof of theorem 4

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. •

9.8 Proof of theorem 5

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 $L_{M_2 N_i}$ 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 $L_{M_2 N_i}$ 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 $L_{M_2 N_i}$, 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 $L_{M_1N_i}$ 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 $L_{M_2N_i}$ 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 $1b$, 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 1c 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. •

9.9 Proof of theorem 6

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. •

9.10 Proof of theorem 7

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.. •

Acknowledgments

Nadia Pisanti were supported by grants from ACI IMPBIO 2003 (project Evol-Rep) and computations were achieved thank to grants from ANR-06-CIS (project PROTEUS).

Disclosure statement

No competing financial interests exist.

References

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

Bouthinon, D. and Soldano, H., 1999. A new method to predict the consensus secondary structure of a set of unaligned rna sequences. *Bioinformatics* 15, 785–798.

Boutonnet, N. S., Rooman, M. J., Ochagavia, M. E., Richelle, J., and Wodak, S. J., 1995. Optimal protein structure alignments by multiple linkage clustering: application to distantly related proteins. *Protein Eng* 8, 647–62.

Brenner, S. E., Koehl, P., and Levitt, M., 2000. The astral compendium for protein structure and sequence analysis. *Nucleic Acids Res* 28, 254–6.

Brint, A. and Willett, P., 1987. Algorithms for the identification of three-dimensional maximal common substructures. *J Chem Inf Comput Sci* 27, 152–8.

Brown, N. P., Orengo, C. A., and Taylor, W. R., 1996. A protein structure comparison methodology. *Computers and Chemistry* 20, 359–380.

Carpentier, M. and Pothier, J., 2007. Protein pairwise structural comparison methods: a review. In de Brevern, A. G., ed., *Recent Advances in Structural Bioinformatics*. Research Signpost, India.

Cook, D. J. and Holder, L. B., 1994. Substructure discovery using minimum description length and background knowledge. *Journal of Artificial Intelligence Research* 1, 231–55.

Crandell, C. W. and Smith, D. H., 1983. Computer-assisted examination of compounds for common three-dimensional substructures. *J. Chem. Inf. Comput. Sci.* 23, 186–97.

Dror, O., Benyamini, H., Nussinov, R., and Wolfson, H., 2003a. Mass: multiple structural alignment by secondary structures. *Bioinformatics* 19 Suppl. 1, i95–104.

Dror, O., Benyamini, H., Nussinov, R., and Wolfson, H. J., 2003b. Multiple structural alignment by secondary structures: algorithm and applications. *Protein Sci* 12, 2492–507.

Eidhammer, I., Jonassen, I., and Taylor, W. R., 2000. Structure comparison and structure patterns. *J Comput Biol* 7, 685–716.

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

Escalier, V., Pothier, J., Soldano, H., and Viari, A., 1998. Pairwise and multiple identification of three-dimensional common substructures in proteins. *J Comput Biol* 5, 41–56.

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

Feng, J., Parida, L., and Zhou, R., 2005. Protein folding trajectory analysis using patterned clusters. In *Proceedings of 3rd Asia-Pacific Bioinformatics Conference, 17-21 January 2005, Singapore*, 95–104.

Gerstein, M. and Altman, R., 1995. Using a measure of structural variation to define a core for the globins. *Comput. Appl. Biosci.* 11, 633–644.

Gerstein, M. and Levitt, M., 1996. Using iterative dynamic programming to obtain accurate pairwise and multiple alignments of protein structures. In *Proc Int Conf Intell Syst Mol Biol*, volume 4, 59–67.

Gerstein, M. and Levitt, M., 1998. Comprehensive assessment of automatic structural alignment against a manual standard, the scop classification of proteins. *Protein Sci* 7, 445–56.

Guda, C., Scheeff, E. D., Bourne, P. E., and Shindyalov, I. N., 2001. A new algorithm for the alignment of multiple protein structures using monte carlo optimization. In *Pacific Symposium on Biocomputing Pacific Symposium on Biocomputing*, 275–86.

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

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

Kawabata, T., 2003. Matras: A program for protein 3d structure comparison. *Nucleic Acids Res* 31, 3367–9.

Koch, I., Kaden, F., and Selbig, J., 1992. Analysis of protein sheet topologies by graph theoretical methods. *Proteins* 12, 314–23.

Koch, I., Lengauer, T., and Wanke, E., 1996. An algorithm for finding maximal common subtopologies in a set of protein structures. *J Comput Biol* 3, 289–306.

Konagurthu, A., Whisstock, J., Stuckey, P., and Lesk, A., 2006. Mustang: a multiple structural alignment algorithm. *Proteins* 64, 559–74.

Leibowitz, N., Fligelman, Z. Y., Nussinov, R., and Wolfson, H. J., 1999. Multiple structural alignment and core detection by geometric hashing. *Proc Int Conf Intell Syst Mol Biol* 169–77.

Leibowitz, N., Fligelman, Z. Y., Nussinov, R., and Wolfson, H. J., 2001. Automated multiple structure alignment and detection of a common substructural motif. *Proteins* 43, 235–45.

Lesk, A. M. and Fordham, W. D., 1996. Conservation and variability in the structures of serine proteinases of the chymotrypsin family. *J Mol Biol* 258, 501–37.

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

Lupyan, D., Leo-Macias, A., and Ortiz, A. R., 2005. A new progressive-iterative algorithm for multiple structure alignment. *Bioinformatics* .

Marin, A., Pothier, J., Zimmermann, K., and Gibrat, J. F., 2002. Frost: a filter-based fold recognition method. *Proteins* 49, 493–509.

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

Murzin, A. G., Brenner, S. E., Hubbard, T., and Chothia, C., 1995. Scop: a structural classification of proteins database for the investigation of sequences and structures. *J Mol Biol* 247, 536–40.

Ochagavia, M. and Wodak, S., 2004. Progressive combinatorial algorithm for multiple structural alignments: Application to distantly related proteins. *Proteins: Structure, Function, and Bioinformatics* 55, 436–454.

Ortiz, A. R., Strauss, C. E., and Olmea, O., 2002. Mammoth (matching molecular models obtained from theory): an automated method for model comparison. *Protein Sci* 11, 2606–21.

Parida, L., 2008. *Pattern Discovery in Bioinformatics*. Chapman & Hall.

Parida, L. and Zhou, R., 2005. Combinatorial pattern discovery approach for the folding trajectory analysis of a β-hairpin. *PLoS Computational Biology* 1.

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

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

Pisanti, N., Soldano, H., Carpentier, M., and Pothier, J., 2006. Implicit and explicit representation of approximated motifs. In Iliopoulos, C. S., Park, K., and Steinhoefel, K., eds., *Algorithms in Bioinformatics*, *Texts in Algorithmics*, volume 6, 1–14. College Press.

Sali, A. and Blundell, T. L., 1990. Definition of general topological equivalence in protein structures. a procedure involving comparison of properties and relationships through simulated annealing and dynamic programming. *J Mol Biol* 212, 403–28.

Shindyalov, I. N. and Bourne, P. E., 1998. Protein structure alignment by incremental combinatorial extension (ce) of the optimal path. *Protein Eng* 11, 739–47.

Simons, K. T., Bonneau, R., Ruczinski, I., and Baker, D., 1999. Ab initio protein structure prediction of casp iii targets using rosetta. *Proteins* Suppl 3, 171–6.

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

Su, S., Cook, D. J., and Holder, L. B., 1999. Applications of knowledge discovery to molecular biology: identifying structural regularities in proteins. *Pac Symp Biocomput* 190–201.

Taylor, W. R., May, A. C., Brown, N. P., and Aszodi, A., 2001. Protein structure: geometry, topology and classification. *Reports on Progress in Physics* 64, 517.

Vialette, S., 2004. On the computational complexity of 2-interval pattern matching problems. *Theoretical Computer Science* 312, 223–249.

Wu, T. D., Schmidler, S. C., Hastie, T., and Brutlag, D. L., 1998a. Modeling and superposition of multiple protein structures using affine transformations: analysis of the globins. *Pac Symp Biocomput* 509–20.

Wu, T. D., Schmidler, S. C., Hastie, T., and Brutlag, D. L., 1998b. Regression analysis of multiple protein structures. *J Comput Biol* 5, 585–95.

Ye, J. and Janardan, R., 2004. Approximate multiple protein structure alignment using the sum-of-pairs distance. *J Comput Biol* 11, 986–1000.

Ye, Y. and Godzik, A., 2003. Flexible structure alignment by chaining aligned fragment pairs allowing twists. *Bioinformatics* 19 Suppl 2, II246–II255.

Ye, Y. and Godzik, A., 2005. Multiple flexible structure alignment using partial order graphs. *Bioinformatics* 21, 2362–9.