An Efficient Motif Search Algorithm based on a Minimal Forbidden Patterns Approach

Tarek El Falah, Thierry Lecroq and Mourad Elloumi

Abstract One of the problems arising in the analysis of biological sequences is the discovery of sequence similarity by finding common motifs. Several versions of the *motif finding problem* have been proposed for dealing with this problem and for each version, numerous algorithms have been developed.
In this paper, we propose an exact algorithm, called SMS-H-FORBID to solve the *Simple Motif Problem* (SMP). SMS-H-FORBID is based on clever techniques reducing the number of patterns to be searched for. These techniques are fundamentally different from the ones employed in the literature making SMP more practical.

1 Introduction

The problem of detecting common motifs across a set of strings is a problem of interest to both biologists and computer scientists. The *motif finding problem* consists in finding substrings that are more or less conserved in a set of strings. To have a satisfactory practical solution several versions of the *motif finding problem* have been defined very precisely [3]. Indeed, the general version of this problem is NP-hard [9]. We find in the literature the *Planted (l,d)-Motif Problem* (PMP) [2, 7, 8], the *Extended (l,d)-Motif Problem* (ExMP) [6, 11], the *Edited Motif Problem* (EdMP) [9, 10], and the *Simple Motif Problem* (SMP) [5, 9, 4].

Tarek El Falah
Research Unit of Technologies of Information and Communication, Higher School of Sciences and Technologies of Tunis, 1008 Tunis, Tunisia, and University of Rouen, LITIS EA 4108 e-mail: Tarek.Elfalah@etu.univ-rouen.fr

Thierry Lecroq
University of Rouen, LITIS EA 4108 e-mail: Thierry.Lecroq@univ-rouen.fr

Mourad Elloumi
Research Unit of Technologies of Information and Communication, Higher School of Sciences and Technologies of Tunis, 1008 Tunis, Tunisia e-mail: Mourad.Elloumi@fsegt.rnu.tn

In this paper, we are interested in the same version of the SMP presented in [4]: Let $Y = \{y_0, y_2, \ldots, y_{n-1}\}$ be a set of strings built from an alphabet Σ, $p > 0$ be an integer and $q \leq n$ be a quorum, find all the simple motifs of length at most p that occurs in at least q strings of Y. A *simple motif* has the same definition as in [5, 9]: it is a string built from an alphabet $\Sigma \cup \{?\}$ that cannot begin or end with ?, where Σ is a set of symbols and $? \notin \Sigma$ is a wildcard symbol, it can be replaced by any symbol from Σ.
In [4], we have proposed a new approach to find simple motifs. This approach is based on clever techniques that reduce the number of patterns to be searched for. We have also presented algorithm SMS-FORBID which is a pattern-based algorithm. In this paper, we present an efficient algorithm called SMS-H-FORBID which uses a hash table in order to make easier finding candidate patterns. Moreover, it maintains a set of minimal forbidden patterns that do not occur in at least q strings in order to not search for any pattern that contains a factor that has already been unsuccessfully searched.

We organize the rest of the paper as follows: In section 2, we present some useful definitions and notations. In section 3, we explain the minimal forbidden patterns approach on which is based SMS-FORBID and SMS-H-FORBID algorithms. In section 4, we explain a new approach related to SMP on which is based SMS-H-FORBID. In section 5, we compute the complexity of SMS-H-FORBID. In section 6, we give some details about the implementation and present some experimental results. In section 7, we make a conclusion to this paper.

2 Definitions and notations

A *simple motif* is a string built from an alphabet $\Sigma \cup \{?\}$ that cannot begin or end with ?, where Σ is a set of symbols and $? \notin \Sigma$ is a wildcard symbol, it can be replaced by any symbol from Σ. Symbols of Σ are said to be solid while the wildcard symbol ? is said to be non-solid. The length of a simple motif is the number of the symbols that constitute this motif, including the wildcard symbols.

A string of ℓ symbols from Σ is called a ℓ-mer. A string of ℓ symbols from $\Sigma \cup \{?\}$ is called a ℓ-pattern. A ℓ-pattern z_1 is equivalent to a ℓ-pattern z_2 ($z_1 \cong z_2$), if a position in z_2 contains the wildcard symbol or the same position in z_1 contains the wildcard symbol and if a position in z_2 contains a solid symbol then at the same position in z_1 there is the same symbol.

Formally, $z_1 \cong z_2$ if for $1 \leq i \leq \ell$: $\begin{cases} z_1[i] = z_2[i] & \text{or} \\ z_1[i] = ? & \text{or} \\ z_2[i] = ? \end{cases}$

A ℓ-pattern z_1 is more general than a ℓ-pattern z_2 if a position in z_2 contains the wildcard symbol implies that the same position in z_1 contains the wildcard symbol and if a position in z_2 contains a solid symbol then at the same position in z_1 there could be either the same symbol or a wildcard symbol. Formally $z_2[i] = ? \Rightarrow z_1[i] = ?$ and $z_2[i] = a \Rightarrow z_1[i] = a$ or $z_1[i] = ?$ for $1 \leq i \leq \ell$ and $a \in \Sigma$.

Let $Y = \{y_0, y_1, \ldots, y_{n-1}\}$ be a set of strings built from an alphabet Σ and let $N = \sum_{i=0}^{n-1} |y_i|$.

3 Minimal forbidden patterns approach

A pattern z of length at most p is said to be a minimal forbidden pattern if it occurs in less than q strings but all its proper factors beginning and ending with a solid symbol occur in at least q strings.

We have proposed algorithm SMS-FORBID based on a new approach to find simple motifs. The algorithm together with all the different data structures have been presented in details in [4]. The inputs of the algorithm are the set Y of n strings, a quorum $q \leq n$ and an integer p. The algorithm outputs the set of motifs of length at most p that occurs in at least q strings.

Contrary to the algorithm presented in [9], the new approach does not search for all the ℓ-patterns generated from the n strings of Y but it begins by searching the more specific patterns i.e. the less general patterns which avoids the sorting step. Moreover it maintains a set of minimal forbidden patterns that do not occur in at least q strings in order to not search for any pattern that contains a factor that has already been unsuccessfully searched.

The general approach is as follows: For each position on the input strings, we use all the windows of length ℓ for $3 \leq \ell \leq p$. Each window defines an ℓ-mer. Each ℓ-mer x defines a set of ℓ-patterns X. At each position of each ℓ-pattern z of X, the symbol of z is either the symbol at the same position of x or the wildcard symbol except for the first and the last symbols of z that are necessarily non-wildcard symbols. Formally,

$$z[i] = \begin{cases} x[i] \\ \text{or} \\ ? \end{cases}$$

for $1 \leq i \leq \ell - 2$ and $z[0] = x[0]$ and $z[\ell-1] = x[\ell-1]$.

These ℓ-patterns together with the generality relation form a lattice. The minimal element of the lattice is x itself and the maximal element is $x[0]?^{\ell-2}x[\ell-1]$.

Each node of the lattice represents an ℓ-pattern.

The ℓ-patterns are scanned by doing a breadth-first search of the lattice beginning from the minimal element.

When a ℓ-pattern z is considered, if:

- it has already been output or
- it contains minimal forbidden patterns as factors or
- it is more general than an output pattern

then it is disregarded otherwise it is searched in the strings of Y. Then if it occurs in at least q strings it is output and all its successors in the lattice are not considered since they are more general. On the contrary if it does not occur in at least q strings it is added to the set of minimal forbidden patterns.

The generation of the ℓ-patterns is performed using a breadth-first search of the lattice for the following reason. When a ℓ-pattern is discovered all its successors in the lattice, that are more general, do not have to be considered. They are thus marked using a depth-first search of the lattice from the discovered ℓ-pattern. During the remaining of the breadth-first search, marked ℓ-patterns are not considered.

Algorithm SMS-FORBID is of complexity $O(N2^p|\Sigma|^p(p+m))$ in computing time, where m is the maximal length of the sequences of Y. The space complexity of SMS-FORBID is $O(N+2^p+|\Sigma|^p)$.

Next, we present SMS-H-FORBID algorithm.

4 Another approach: SMS-H-Forbid

In order to easily find the candidate patterns we define a table H for every couple of solid symbols and every integer k from 0 to $p-3$ as follows:

$$H[a,b,k] = \{(i,j) \mid y_i[j] = a \text{ and } y_i[j+k+2] = b\}.$$

When a candidate ℓ-pattern is generated from position j in string y_i, if

- it has not already been output or
- it does not contain minimal forbidden patterns as factors or
- it is not more general than an output pattern

its potential occurrences are only searched at the positions in $H[y_i[j], y_i[j+\ell-1], \ell-2]$.

In practice, the elements of $H[a,b,k]$ are sorted in decreasing order of the index of the strings.

The main algorithm is depicted in Fig.1. It builds the set *Res* of searched motifs of length at most p contained in at least q strings and uses a set $\mathscr{T}$ of minimal patterns that are not contained in at least q strings. SMS-H-FORBID scans the strings of Y in the same order than algorithm SMS-FORBID. The breadth-first-search is performed in the same manner as SMS-FORBID [4]. The only changes appear for counting the number of strings containing an ℓ-pattern x generated from y_j (see algorithm COUNT in Fig. 2). The occurrences of x are searched using the list of pairs in $H[x[0], x[\ell-1], \ell-3]$ (see algorithm SEARCH in Fig. 3). Furthermore those pairs (ind, pos) are sorted in decreasing order thus only pairs where $ind > j$ are considered.

5 SMS-H-Forbid Complexities

The algorithm SMS-H-FORBID given in Fig. 1 builds the H table in time $O(Np)$.

```
SMS-H-FORBID($Y, n, p, q$)
 Set every positions of H to $\emptyset$
for $i \leftarrow 0$ to $n-1$ do
    for $j \leftarrow 2$ to $|y_i|-1$ do
        for $k \leftarrow 0$ to $p-3$ do
            $H[y_i[j-k-2], y_i[j], k] \leftarrow H[y_i[j-k-2], y_i[j], k] \cup \{(i, j-k-2)\}$
$Res \leftarrow \emptyset$
$\mathscr{T} \leftarrow \emptyset$
for $j \leftarrow 1$ to $n-q+1$ do
    for $i \leftarrow 1$ to $|y_j|-2$ do
        for $\ell \leftarrow 3$ to $\min\{p, |y_j|-i\}$ do
            for $k \leftarrow 0$ to $\lfloor \ell/2 \rfloor$ do
                BREADTH-FIRST-SEARCH(
                $\triangleright\, y_j[i \,..\, i+\ell-1], 2, q, j)$
return $Res$
```

Fig. 1 The main algorithm.

```
COUNT($x, Y, j, i, \ell$)
$L \leftarrow H[x[0], x[\ell-1], \ell-3]$
$k \leftarrow 1$
Unmarked all strings
while $L \neq \emptyset$ do
    $(ind, pos) \leftarrow$ first element of $L$
    DEQUEUE($L$)
    if $ind \leq i$ or $k + ind - i < q$ then
        return $k$
    else if $y_{ind}$ is not marked then
        if SEARCH($x[1 \,..\, \ell-2], y_{ind}[pos+1 \,..\, pos+\ell-2], \ell-2$) then
            mark $y_{ind}$
            $k \leftarrow k+1$
            if $k \geq q$ then
                return $k$
```

Fig. 2 Count the number of strings of Y that contain motif x.

```
SEARCH($x, y, \ell$)
for $i \leftarrow 0$ to $\ell-1$ do
    if $x[i] \neq ?$ and $x[i] \neq y[i]$ then
        return FALSE
return TRUE
```

Fig. 3 Search if x is equivalent to y.

The algorithm SMS-H-FORBID given Fig. 1 scans all the positions of the n sequences of Y. For each position it considers all the ℓ-patterns defined by the corresponding ℓ-mer for $3 \leq \ell \leq p$. The number of elements of all the corresponding lattices is bounded by 2^{p+1}.

Processing one ℓ-pattern x (see algorithm COUNT in Fig. 2) consists in:

1. looking if x is in *Res*;
2. checking if x contains minimal forbidden patterns;
3. searching x in the n sequences of Y using the H table.

Looking if x is included in *Res* can be done in $O(|x|)$ time using a trie for *Res*.

Checking if x contains minimal forbidden patterns consists in using an algorithm for searching a single pattern with wildcard symbols in a text with wildcard symbols for every pattern in $\mathscr{T}$. This can be done in $O(|\mathscr{T}||x|)$.

The search of one ℓ-pattern x in the strings of Y (see algorithm SEARCH in Fig. 3) consists in checking all the pairs in $H[x[0], x[\ell-1], \ell-3]$. Thus the time complexity of algorithm SEARCH is $O(\ell N)$.

Altogether the time complexity of the algorithm SMS-H-FORBID is $O(N2^p|\Sigma|^p(pm))$ where m is the maximal length of the sequences of Y.

The space complexity of the H table is $O(|\Sigma|^2 p)$.

The algorithm requires to build and traverse all the lattices corresponding to ℓ-patterns. An array of size $2^{\ell-2}$ is used to mark the nodes of each lattice. Thus the space complexity for the lattices is $O(2^p)$.

In the worst case the size of *Res* and $\mathscr{T}$ is bounded by $|\Sigma|^p$.

Altogether the space complexity of the algorithm SMS-H-FORBID is $O(|\Sigma|^2 p + 2^p + |\Sigma|^p)$.

In practice $|\Sigma|^2 p < N$.

6 Experimental study

We have implemented SMS-FORBID and SMS-H-FORBID algorithms in C on a pentium 4, 2 GHz machine with 3 GB RAM.
We have measured the computing time of these algorithms, for different values of q, N and p, on random strings of length 100 built on DNA alphabet and also on protein alphabet. The curves given in Fig. 4 and Fig. 5 show, respectively, the results obtained for SMS-H-FORBID with $p = 7$ on DNA alphabet and the variation of the computing time t in function of q for SMS-H-FORBID with $p = 5$ on protein alphabet. The X-axis represents the quorum q and the Y-axis represents the computing time t. All these results have been obtained through computing an average on 15 draws.

As we can notice, the curve given in Fig. 4 has a bell-like shape. We can explain this as follows: Indeed, for a low value of the quorum, the possibility of finding quickly the motifs is higher and the algorithm will not make unnecessary search for them in the rest of the strings. When the quorum is getting closer to N, the number

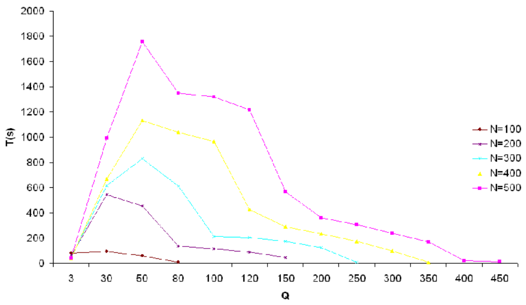

Fig. 4 Computing time of SMS-H-FORBID in function of q and N for $p = 7$ on DNA alphabet

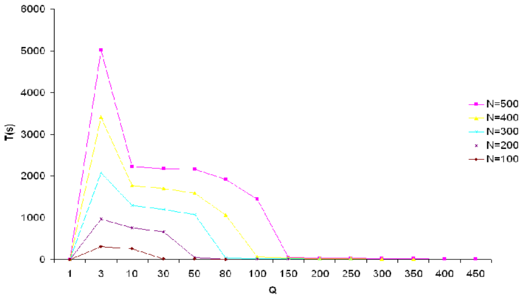

Fig. 5 Computing time of SMS-H-FORBID in function of q and N for $p = 5$ on protein alphabet

of detected minimal forbidden motifs increases. Then, the possibility that one of these motifs appears in the current window is higher. Hence, the possibility for not comparing the current window with the substrings of the other input strings is also higher. So, the possibility to reduce the computing time is also higher.
It is also remarkable to note that the curve given in Fig. 5 has also a bell-like shape. However, the peak of the curve is for a low value of the quorum. Indeed, the size of the protein alphabet lets that the possibility of finding simple motifs that have occurrences in at least q strings decreases rapidly as q increases.

Concerning real biological data, we are experimenting our algorithms on various protein sequences. The first results seem to be interesting, i.e., the extracted motifs are more specific, and the computing time and the memory space are reduced. The table below shows the computing time of SMS-H-Forbid for some protein families for $p = 7$ and $q = 10$.

Protein family	**N**	**Average Length**	**Computing time(s)**
dehydrogenase	25	324	346.875
phospholase	20	305	160.563
sam domain	19	434	338.09
yjgpyjgq	36	358	1495.719

7 Conclusion

To have a practical solution for SMP, we have introduced the notion of generality between patterns. The main purpose of this notion is to reduce the number of motifs to be considered by eliminating similar or inferior ones. The general approach is based on maintaining a set of minimal forbidden patterns that do not occur in at least q sequences in order to not search for any pattern that contains a factor that has already been unsuccessfully searched.

In this paper, we developed an efficient algorithm well performing in practice by reducing the number of patterns to be searched for. In fact, SMS-H FORBID finds the more specific motifs and so that identifies important motifs for biologists.

As improvement, we have to develop an algorithm which performs multiple pattern matching with wildcards. Moreover, suffix trees used in SMS-FORBID [4] are space consuming thus suffix arrays or even BWT [1] can be a good alternative to save space.

In the future, we will study how to determine an approximate algorithm to have a faster solution for SMP.

References

1. D. Adjeroh, T. Bell, and A. Mukherjee. *The Burrows-Wheeler Transform*. Springer, 2008.
2. F. Y. L. Chin and H. C. M. Leung. Voting algorithm for discovering long motifs. In *Proceedings of Asia-Pacific Bioinformatics Conference*, pages 261–272, 2005.
3. T. El Falah, M. Elloumi, and T. Lecroq. Motif finding algorithms in biological sequences. In *Algorithms Computational Molecular Biology: Techniques, Approaches and Applications, Wiley Book Series on Bioinformatics: Computational Techniques and Ingeneering*, pages 387–398. Wiley-Blackwell, John Wuley and Sons Ltd., New Jersey, USA, 2011.
4. T. El Falah, T. Lecroq, and M. Elloumi. Sms-forbid: an efficient algorithm for simple motif problem. In *Proceedings of the ISCA 2nd International Conference on Bioinformatics and Computational Biology*, pages 121–126, Honolulu, Hawai, 2010.
5. A. Floratos and I. Rigoutsos. On the time complexity of the teiresias algorithm. Technical report, Research Report RC 21161 (94582), IBM T.J. Watson Research Center, 1998.
6. H. C. M. Leung and F. Y. L. Chin. An efficient algorithm for the extended (l,d)-motif problem, with unknown number of binding sites. In *Proceedings of the Fifth IEEE Symposium on Bioinformatics and Bioengineering (BIBE'05)*, pages 11–18, 2005.
7. A. Price, S. Ramabhadran, and P. A. Pevzner. Finding subtle motifs by branching from sample strings. *Bioinformatics*, 1(1):1–7, 2003.
8. S. Rajasekaran, S. Balla, and C. H. Huang. Exact algorithms for planted motif challenge problems. *Journal of Computational Biology*, 12(8):1117–1128, 2005.
9. S. Rajasekaran, S. Balla, C.-H. Huang, V. Thapar, M. Gryk, M. Maciejewski, and M. Schiller. High-performance exact algorithms for motif search. *Journal of Clinical Monitoring and Computing*, 19:319–328, 2005.
10. M. F. Sagot. Spelling approximate repeated or common motifs using a suffix tree. In *In C. L. Lucchesi and A. V. Moura, editors, LATIN'98: Theoretical Informatics, volume 1380 of Lecture Notes in Computer Science, Springer-Verlag*, pages 111–127, 1998.
11. M.P. Styczynski, K. L. Jensen, I. Rigoutsos, and G.N. Stephanopoulos. An extension and novel solution to the (l,d)-motif challenge problem. *Genome Informatics*, 15(2):63–71, 2004.