A Comparative Study of Bases for Motif Inference[*]

Nadia Pisanti [†] Maxime Crochemore [‡] Roberto Grossi [§]
Marie-France Sagot [¶]

June 14, 2004

Abstract

Motif inference is at the heart of several time-demanding computational tasks, such as in molecular biology, data mining and identification of structured motifs in sequences, and in data compression, to name a few. In this scenario, a motif is a pattern that appears repeated at least a certain number of times (the quorum), to be of interest. The pattern can be approximated in that some of its characters can be left unspecified (the don't cares). Motif inference is not aimed at searching a given pattern but, rather, at discovering all the possible patterns that appear as motifs in the given input string. The combinatorial explosion of these patterns makes their discover an exponential-time computation. For this, the notion of basis has been recently introduced to succinctly represent all of them within reasonable time and space bounds. The goal of the paper is to shed light on the state of the art for this emerging field and to add further properties to what is currently known.

1 Introduction

Motivations and Applications

Pattern inference in the sense of finding regularities in various types of data is a very old activity which has generated much research in the past. The literature has been vast on the topic, the interested reader may consult for instance [5] for more details. Most approaches

[*] The work described in this paper is partially supported by the French program bioinformatique EPST 2002 "Algorithms for Modelling and Inference Problems in Molecular Biology".

[†] Dipartimento di Informatica, Università di Pisa, Italy (pisanti@di.unipi.it). Work partially supported by the Italian MIUR project PRIN "ALINWEB: Algorithmics for Internet and the Web".

[‡] Institut Gaspard-Monge, University of Marne-la-Vallée, 77454 Marne-la-Vallée CEDEX 2, France and King's College London (Maxime.Crochemore@univ-mlv.fr). Work partially supported by CNRS, NATO Science Programme grant PST.CLG.977017, and Wellcome Trust Foundation.

[§] Dipartimento di Informatica, Università di Pisa, Italy (grossi@di.unipi.it). Work partially supported by the Italian MIUR project PRIN "ALINWEB: Algorithmics for Internet and the Web".

[¶] INRIA Rhône-Alpes, Laboratoire de Biométrie et Biologie Évolutive, Université Claude Bernard, 69622 Villeurbanne cedex, France (Marie-France.Sagot@inria.fr). Work partially supported by CNRS-INRIA-INRA-INSERM action BioInformatique and Wellcome Trust Foundation.

have been based on the use of statistics to infer the desired information. Combinatorial methods have thus been rare. Pattern inference in textual data is a sub-topic that has been also much explored already. Among the more theoretical work in this area, there has been a large number of papers on identifying and characterizing periodicity and repeats in a string. One may consult for instance [7] or [8]. Much of this theoretical work has been concerned however with exact repeats, unlike the more applied work discussed in [5].

In this paper, we address the problem of pattern inference in strings, with some clear applications in mind that require working with approximate regularities. We are therefore interested in finding approximate repeats in a text. We are concerned also with the combinatorial type of approaches for detecting such repeats.

The applications we have in mind can be varied. The main one which initially motivated the work is computational biology. In computational biology, approximate repeats may correspond to biologically functional units. Such units may represent, for instance, protein-DNA or protein-RNA binding sites, that is fragments of a molecule of DNA, RNA or protein that will come into interaction with another macromolecule in order for a function to happen. The molecular fragment may be seen as a word in a text (the whole molecule) that must be enough conserved through evolutionary time for the recognition of the two molecules by each other to take place. The conservation is in general not strict, hence the need to model it by introducing approximate repeats. Other applications, different from biology, present roughly the same characteristics. In document duplication and plagiarism for instance, common contents are detected in a set of documents divided into semantic units such as paragraphs or pages. It is possible to represent the distinct (stemmed or categorized) words in each unit by bit vectors as in the vector model used for information retrieval described in [3]. Paragraphs with similar contents are filtered for further processing since they share many common bits giving rise to approximate patterns in the concatenation of the bit vectors. In data mining systems [6] also, researchers have realized that discovering repetitions called "frequent itemsets" are at the heart of the problem of finding association rules. We can use for this purpose the same approach with bit vectors as above, each bit vector representing an itemset. The association rules can be based not only upon items appearing together *often enough*, but also upon their relation with the items that frequently do *not* appear (the latter corresponding to a sort of negative association rule). Motifs with don't cares are also useful in system security, where an intrusion is often indicated by a sequence of frequent operations appearing intermixed with others. Also in this case, we can break the log file of the operations into units and perform approximate pattern discovery. Finally, very recently, data compression has also started to use repeated approximate patterns of a special type [2]. These are patterns with don't cares where a don't care is a symbol that matches anything.

All of the above examples are some of the best-known formulations that address the common issue of detecting approximate repeated patterns in several different contexts, thus revealing the algorithmic relevance of the problem.

Unfortunately, once approximate patterns are considered, the complexity of the algorithms for their discovery may easily become exponential due to the potentially exponential growth in the number of such patterns that may appear repeated in a string. Consider for instance the string $\texttt{A}\cdots\texttt{ATA}\cdots\texttt{A}$ (same number of As on both sides of the letter T). It contains many repetitions of As intermixed with don't cares. In general, strings over a small alphabet, such as some "real-life" occurring ones like DNA sequences, will contain an exponential number of repeats. A few heuristics try to alleviate this problem by reducing the number of interesting repeated patterns they identify to make feasible any further processing of them, but they often cannot guarantee sub-exponential bounds in the number of repeats in the worst case [9].

About this paper

In this paper, we describe the algorithmic ideas behind pattern discovery while getting some insight into their combinatorial complexity. Although more general (and computationally demanding) notions of approximate patterns are possible [4, 16, 17], the discovery and extraction of patterns with don't cares is flexible enough for several applications. These are therefore the types of patterns we shall consider in this paper.

Given a pattern x for a string s of length n, let us denote the set of starting positions of occurrences of x on s by $\mathcal{L}_x \subseteq [0\,..\,n-1]$, and by $|\mathcal{L}_x|$ is the number of occurrences of x. We define a pattern to be a motif when $|\mathcal{L}_x| \geq q$, that is, the pattern appears at least q times in a string, where q is called the *quorum* the pattern must satisfy to be of any interest. We focus our attention on *maximal* motifs x, informally characterized as satisfying $|\mathcal{L}_x| \neq |\mathcal{L}_y|$ for any other motif y more *specific* than x, that is, obtained from x by extending x, to the left or right, with don't cares and at least one alphabet letter and/or by replacing one or more of the don't cares in x with alphabet letters. In other words, x appears in y but x occurs in s more times than y does, which is considered informative for discovering the repetitions in s. For example, $x = \texttt{M}\circ\circ\texttt{T}\circ\texttt{E}$ is maximal in $\texttt{COMMITTEE}$ for $q=2$ (and $\mathcal{L}_x = \{2, 3\}$) while $z_1 = \texttt{M}\circ\circ\circ\circ\texttt{E}$ and $z_2 = \texttt{T}\circ\texttt{E}$ are not maximal since x is more specific than both of them, and has the same number of occurrences ($\mathcal{L}_{z_1} = \mathcal{L}_x$ and $\mathcal{L}_{z_2} = \{5, 6\}$). Maximality is intuitively relevant since each maximal motif x indirectly represents all non-maximal motifs z that are less specific than it, namely, such that $\mathcal{L}_z = \mathcal{L}_x + d$, where $\mathcal{L}_x + d$ denotes the set $\{i + d \mid i \in \mathcal{L}_x\}$ of occurrences found exactly d positions apart from those in $\mathcal{L}_x$. Unfortunately, this property is not enough to significantly bound the number of maximal motifs. For example, $\texttt{A}\cdots\texttt{ATA}\cdots\texttt{A}$ contains an exponential number of maximal motifs for $q = 2$ (see Section 2.1).

A further requirement on the maximal motifs relies on eliminating those whose occurrences can be derived from the occurrences of other motifs. The motifs selected form a *basis* for the maximal motifs, in that they can generate all the other maximal motifs by simple mechanical rules. They can thus be expressed mathematically in the algebraic sense of the term. This elegant introduction of a mathematical well-defined concept to the problem of motif discovery was, to the best of our knowledge, first introduced in the literature by Parida *et al.* [9] for applications to molecular biology. It remains unclear what is the upper bound for the initial basis described in [9] for a quorum of 2. This initial work inspired a number of others which introduced new definitions of bases of smaller sizes [13] [11] [14] [15] [12] [2]. The main motivation for this later work was computational biology except for [2] where the area of application was for data compression. Pisanti *et al.*, started exploring in [14, 15] the relation between Parida's initial basis and their own. The main purpose of the current paper is to explore further this comparison and extend it to the other related work in this scenario. The paper therefore makes an in-depth comparative exploration of the combinatorial properties of the various variants of bases for motifs and of the accompanying algorithms for identifying such bases. As a result, small differences in some basic definitions can definitively change the computational complexity of the corresponding bases, in terms of both time and space bounds. The paper also investigates new directions for possible extensions of the notion of a basis for some special situations (*e.g.* when the input is two strings).

The paper is organized as follows. Section 2 contains an overview of all definitions of a basis of motifs that have appeared in the literature, along with some of their common properties. Section 3 introduces the notions and other properties that are specific to some of the definitions. Section 4 shows lower and upper bounds on the size of the various bases for the special case of quorum $q = 2$. It also mentions a couple of results that have appeared in the literature concerning algorithms for computing bases. Section 5 contains some results that give for any quorum q a characterization of one of the defined bases, called the basis of tiling motifs, in terms of that for smaller quorum. Section 5 shows also that for increasing values of the quorum q, a lower bound on the size of all bases grows exponentially. Finally, Section 6 introduces two possible extensions of the notion of tiling motifs to the case of motifs common to two input strings, as well as efficient algorithms for computing them.

2 Bases of Motifs

This section contains some basic notions and terminology concerning motifs and lists all the definitions of a basis of motifs that have appeared in the literature. We detail the differences among the definitions that lead to different space and time bounds as we shall see in the rest of the paper. Examples are included in order to make the presentation clear.

The length of a string t is denoted by $|t|$ and the letter at position i of t by $t[i]$. Therefore, $t = t[0]t[1] \cdots t[|t| - 1]$. The input string is a word of length n over an alphabet Σ. The letters of Σ are called *solid characters* to distinguish them from an additional letter not in Σ called *don't care* and denoted by $\circ$. A don't care matches any other letter. The motifs in the input string correspond to the repetitions with don't cares that appear in the string.

Definition 1 (Pattern) *A* pattern *is a string in* $\Sigma \cup \Sigma(\Sigma \cup \{\circ\})^*\Sigma$*, that is, a string on the alphabet* $\Sigma \cup \{\circ\}$ *that starts and ends with a solid character.*

We also define a notion of partial order among elements of the set $\Sigma \cup \{\circ\}$ that we later extend to words.

Definition 2 (Specifity) *The* specificity relation *denoted by* $\preceq$ *is defined on* $\Sigma \cup \{\circ\}$ *as follows:* $\sigma_1 \preceq \sigma_2$ *if and only if* $\sigma_1 = \sigma_2$ *or* $\sigma_1 = \circ$*, for* $\sigma_1, \sigma_2 \in \Sigma \cup \{\circ\}$*.*

Strings are implicitly padded to the left and to the right with an infinite number of don't cares. In other words, for a string t, we consider that $t[j] = \circ$ for any integer j such that $j < 0$ or $j \geq |t|$. In this way the relation $\preceq$ extends to strings as well: for any two strings $u, v \in (\Sigma \cup \{\circ\})^*$, we have $u \preceq v$ if and only if $u[j] \preceq v[j]$ for any integer j. We refer to the condition $u \preceq v$ saying that u is *less specific* than v.

Definition 3 (Occurrence) *Given two strings* $u, v \in (\Sigma \cup \{\circ\})^*$*, we say that* u occurs at position ℓ in v *if and only if* $u[j] \preceq v[j + \ell]$ *for any integer* j*.*

Given a string $s \in \Sigma^*$, with $|s| = n$, we define the motifs as special cases of patterns in s.

Definition 4 (Motif) *For a pattern x, the* location list $\mathcal{L}_x$ *contains the positions of all the occurrences of x in s. Given a special parameter $q \geq 2$, called* quorum*, we say that pattern x is a* motif *(according to s and q) if and only if $|\mathcal{L}_x| \geq q$.*

Example Let $s = \texttt{CACACATACATAC}$ and $q = 2$. Pattern $x_1 = \texttt{CA}\circ\texttt{ACATAC}$ is a motif with occurrence list $\mathcal{L}_{x_1} = \{0, 4\}$, and pattern $\texttt{ACA}\circ\texttt{A}$ is also a motif with occurrence list $\{1, 3\}$. On the other hand, the pattern $\texttt{T}\circ\texttt{CA}$ is not a motif because it occurs only once.

Let us fix from now on the input string $s \in \Sigma^n$. We are interested in finding all the motifs of s, that is, all patterns that appear at least q times in s. Unless differently specified, we assume that the quorum is $q = 2$. The problem of finding all the motifs is computationally difficult, since their number is possibly exponential in the worst case. We therefore define a few concepts that allow to reduce this quantity, leading to the various notions of bases.

2.1 Maximal Motifs

The first concept we discuss is the *maximality* of a motif, which is shared by all basis definitions. Intuitively, a motif x is maximal if no other motif y is more specific than it (i.e., y has a solid character at a position where x has a don't care, or it is longer) *and* simultaneously has the same occurrences as x, possibly shifted by some positions.

Given a set $\mathcal{L}$ of occurrences, we denote by $\mathcal{L} + d$ the set $\{i + d \mid i \in \mathcal{L}\}$, where d is any given integer (it may be negative).

Definition 5 (Maximality) *A motif x is* maximal *when, for any motif y and integer d such that $\mathcal{L}_y = \mathcal{L}_x + d$, we have that y occurs in x at position $d \geq 0$.*

Example (continued) Motif $x_1 = \texttt{CA}\circ\texttt{ACATAC}$ of the previous example is maximal because if it is extended or if its don't care symbol is replaced by a solid character, then it loses at least one of its occurrences. On the contrary, motif $\texttt{CACA}\circ\texttt{A}$ with list $\{0, 2\}$ is not maximal since it occurs in $x_2 = \texttt{CACA}\circ\texttt{A}\circ\texttt{A}\circ\texttt{A}$ which has the same occurrences. Indeed, x_2 is maximal and it is an extension of $\texttt{CACA}\circ\texttt{A}$.

Definition 5 is an equivalent formulation of the one given in [9] and follows from the examples given there. It is known from [9] that there can still be an exponential number of maximal motifs in a string. An example is given for quorum $q = 2$ with the string $t_k = \texttt{A}^k\texttt{TA}^k$ of length $2k + 1$ which has a number of maximal motifs exponential in k. This can be shown by first noticing that any motif of t_k can contain only $\texttt{A}$ as a solid character because no motif containing $\texttt{T}$ can occur twice. Secondly, let us consider, for all $1 \leq i \leq k - 2$, the set of motifs X_i with all the motifs of length k containing exactly i don't cares. Each such motif occurs $i + 2$ times in the string, one for each distinct don't care matching the letter $\texttt{T}$, and two occurrences that match the first and the last k letters of t_k. Moreover, all these motifs are maximal. In fact, any extension to the left or to the right would cause the loss of at least one among the two last occurrences mentioned above. Similarly, any replacement of a don't care with an $\texttt{A}$ would cause the loss of one of the remaining i occurrences. Since we have that $|X_i| = \binom{k-2}{i}$, and since $i \in [1, k - 2]$, then there are $\sum_{i=1}^{k-2} \binom{k-2}{i} = 2^{k-2}$ such motifs. The fact that $|t_k| = \Theta(k)$ leads to the exponential result.

2.2 Irredundant Motifs

The first notion of a basis of motifs with don't cares was given in [9] and further explored in [10]. It is based on the notion of *irredundant motif* which we formally introduce in this section.

Definition 6 (Irredundant motif) *A maximal motif x is* irredundant *if, for any maximal motifs y_1, y_2, $\ldots$, y_k such that $\mathcal{L}_x = \cup_{i=1}^{k} \mathcal{L}_{y_i}$, motif x must be one of the y_i's. Vice versa, if all the y_i's are different from x, pattern x is said to be* covered *by motifs y_i, y_2, $\ldots$, y_k.*

Example (continued) Consider again the string $s = \texttt{CACACATACATAC}$. Motif $x_2 = \texttt{CACA}\circ\texttt{A}\circ\texttt{A}\circ\texttt{A}$ (which is maximal as we have shown in the previous example) is irredundant because its occurrence list $\mathcal{L}_{x_2} = \{0, 2\}$ cannot be built as the union of lists of maximal motifs different from x_2 itself. Likewise, motif $x_4 = \texttt{CA}\circ\texttt{AC}$ with $\mathcal{L}_{x_4} = \{0, 4, 8\}$ is irredundant as well. It is maximal because an extension to the left/right would cause the loss of the leftmost/rightmost occurrence, and its don't care matches different solid characters in different occurrences. It is irredundant because occurrences at position 0 and 4 can be covered by x_2, but no maximal motif can cover position 8. On the other hand, the motif $x_3 = \texttt{CA}\circ\texttt{A}$ with location list $\mathcal{L}_{x_3} = \{0, 2, 4, 8\}$ is maximal but it is redundant because $\mathcal{L}_{x_3} = \mathcal{L}_{x_2} \cup \mathcal{L}_{x_4}$.

In [9], irredundant motifs are suggested as a generator for all repeated motifs in a string.

Definition 7 ($\mathcal{B}_I$) *Given a string s and a quorum q, the* basis of irredundant motifs *$\mathcal{B}_I$ is the set of all irredundant motifs for s and q. The size of $\mathcal{B}_I$ is the number of motifs it contains.*

Example (continued) The basis of irredundant motifs $\mathcal{B}_I$ for the string s of the previous example has size 7 and it contains the motifs $x_0 = \texttt{AC}$, $x_1 = \texttt{CA}\circ\texttt{ACATAC}$, $x_2 = \texttt{CACA}\circ\texttt{A}\circ\texttt{A}\circ\texttt{A}$, $x_4 = \texttt{CA}\circ\texttt{AC}$, $x_5 = \texttt{ACA}\circ\texttt{A}$, $x_6 = \texttt{A}\circ\texttt{A}$, and $x_7 = \texttt{A}\circ\texttt{A}\circ\texttt{A}$.

The set $\mathcal{B}_I$ is a generator for all maximal motifs in s. In [9, 10], an algorithm is shown which generates all maximal motifs starting from the basis of irredundant motifs. The procedure is based on an iterative application of a *generating operator* $\oplus$ that we define as follows.

Definition 8 ($\oplus$) *Given $\sigma_1, \sigma_2 \in \Sigma \cup \{\circ\}$ with $\sigma_1 \neq \sigma_2$, we have $\sigma_1 \oplus \sigma_2 = \circ$ and $\sigma_1 \oplus \sigma_1 = \sigma_1$. This operator is extended to two padded strings $x, y \in \Sigma^*$ with $|x| = |y|$ in the obvious way: $u = x \oplus y$ is such that $u[i] = x[i] \oplus y[i]$ for all integers i between 0 and $|x| - 1$.*

Example (continued) Let us consider again the input string $s = \texttt{CACACATACATAC}$. We showed that the motif $x_3 = \texttt{CA}\circ\texttt{A}$ is redundant because its occurrence list is the union of those of $x_2 = \texttt{CACA}\circ\texttt{A}\circ\texttt{A}\circ\texttt{A}$ and $x_4 = \texttt{CA}\circ\texttt{AC}$. Indeed, we have that $x_3 = \texttt{CA}\circ\texttt{A} = \texttt{CACA}\circ\texttt{A}\circ\texttt{A}\circ\texttt{A} \oplus \texttt{CA}\circ\texttt{AC} = x_2 \oplus x_4$.

Notice that the $\oplus$ operator is such that, if $u = x \oplus y$ is not the empty word, then $\mathcal{L}_u = \mathcal{L}_x \cup \mathcal{L}_y$. The $\oplus$ operator is also associative (and commutative) and therefore can be extended in an obvious way to three or more strings. Observe finally that Definition 8 is equivalent to the definition of $\oplus$ in [10] which states that $u = x \oplus y$ if u is the most specific pattern such that $u \preceq x, y$.

Each redundant maximal motif x can be written as $y_1 \oplus y_2 \oplus \cdots \oplus y_p$ where the y_i's are irredundant motifs (it is enough to choose the y_i's that cover x according to the Definition 6, possibly iterating the process if these are redundant). Different sets of irredundant motifs $y_1, y_2, \ldots, y_p$ can nevertheless lead to the same x. This makes inefficient the process of exhaustively generating all motifs from the subset of irredundant ones. For this reason, an output sensitive algorithm (in the sense of taking time proportional to the size of the output) is suggested in [10]. The algorithm first builds maximal subsets P of $\mathcal{B}_I$, each containing motifs that start with the same letter and are such that none is less specific than another. It is shown in [10] that it is not necessary to consider generations involving motifs belonging to different subsets P as the resulting maximal motifs may be obtained by generations involving motifs belonging to a same subset. The motifs in each subset are aligned on their first letter. For each column of the alignment, each letter appearing at least q times in the column leads to the generation of a new maximal motif.

2.3 Tiling Motifs

In this section, we describe the notion of a basis for motifs introduced in [13] and further explored in [14, 15].

Definition 9 (Tiling) *A maximal motif x is a* tiling motif *if, for any choice of maximal motifs y_1, y_2, …, y_k and integers d_1, d_2, …, d_k such that $\mathcal{L}_x = \cup_{i=1}^{k}(\mathcal{L}_{y_i} + d_i)$, there exists at least one value of i such that $y_i = x$. Otherwise, motif x is a* tiled motif *and motifs y_i, y_2, …, y_k* tile *x.*

Example (continued) Given $s =$ `CACACATACATAC`, its motif $x_5 =$ `ACA∘A` with $\mathcal{L}_{x_5} = \{1, 3, 7\}$ is maximal, but it is tiled by x_1 and x_2 (where $\mathcal{L}_{x_1} = \{0, 4\}$ and $\mathcal{L}_{x_2} = \{0, 2\}$) because its occurrence list can be obtained as $(\mathcal{L}_{x_1} + 3) \cup (\mathcal{L}_{x_2} + 1)$. Hence, the motif x_5 is not tiling. On the other hand, motif x_1 is tiling.

According to Definition 9, if x is tiled by y_1, y_2, …, y_k for given integers d_1, d_2, …, d_k, then $k > 1$. In fact, for $k = 1$, the equality $\mathcal{L}_x = \mathcal{L}_{y_1} + d_1$ with $x \neq y_1$ would contradict the maximality of x or y_1. Observe also that the y_i's in Definition 9 are not necessarily distinct.

Definition 10 ($\mathcal{B}_T$) *Given a string s and a quorum q, the* basis of tiling motifs *$\mathcal{B}_T$ is the set of all tiling motifs for s and q. The size of $\mathcal{B}_T$ is the number of these motifs.*

Example (continued) The basis $\mathcal{B}_T$ of tiling motifs for the string s of the previous example has size 2 and it contains only the motifs $x_1 =$ `CA∘ACATAC` with $\mathcal{L}_{x_1} = \{0, 4\}$, and $x_2 =$ `CACA∘A∘A∘A` with $\mathcal{L}_{x_2} = \{0, 2\}$.

2.4 Primitive Motifs

In [11, 12], the concept of *primitive motifs* has been introduced as another possible notion of basis for repeated motifs with don't cares. Both in [11] and in [12], two different definitions have been proposed and claimed to be equivalent. The first definition is the following.

Definition 11 (Primitive) *A maximal motif x is* primitive *if and only if there are no motifs y_i (not necessarily maximal) with $y_i \neq x$ for $1 \leq i \leq k$, such that $\mathcal{L}_x = \cup_{i=1}^{k} \mathcal{L}_{y_i}$.*

In other words, the patterns that are candidate to *cover* x do not need to be maximal, but they have to occur at least q times. Hence, any right extension of x, or any motif obtained from x by replacing a don't care with a letter is a candidate for the set of y_i's of Definition 11, provided it has at least q occurrences. Although not explicitely stated in [11, 12] (but clearly meant according to the examples that follow the definition of primitive motif in the paper[1]), it should be further required that the y_i's in Definition 11 do not occur in x, otherwise many motifs would be discarded because they are covered, for instance, by their prefixes occurring as many times as the motif itself (which is in contrast with the requirement of maximality).

The second definition given in [11, 12] is claimed to be equivalent to the first one; rather, it is identical to the definition of tiling motifs (and produced independently). Definition 11 leads to a superset of the set of tiling motifs. Indeed, the fact of not allowing for shifts when trying to cover x (and thus to discard it), leads to keep more motifs than in the case of tiling motifs, as shown in the following example.

Example (continued) Let us consider again the input string $s = \texttt{CACACATACATAC}$ with $q = 2$, and the motif $x_0 = \texttt{AC}$ with $\mathcal{L}_{x_0} = \{1, 3, 7, 11\}$. We have that x_0 is maximal because any extension to the right loses the rightmost occurrence, no one-letter extension to the left can be made with a solid character, and longer left extensions would lose the leftmost occurrence. Using the definition of tiling motifs, x_0 is discarded due to motifs $x_1 = \texttt{CA}\circ\texttt{ACATAC}$ and $x_2 = \texttt{CACA}\circ\texttt{A}\circ\texttt{A}\circ\texttt{A}$ with, respectively, $\mathcal{L}_{x_1} = \{0, 4\}$ and $\mathcal{L}_{x_2} = \{0, 2\}$, because $\mathcal{L}_{x_0} = (\mathcal{L}_{x_1} + 7) \cup (\mathcal{L}_{x_2} + 1)$. On the other hand, using Definition 11, no motif occurring at least twice can cover the occurrence at position 11 without a shift. Indeed, the only candidate different from x_0 itself is $y = \texttt{A}$, but the occurrence list of y is a superset of that of x_0. Hence x_0 is primitive but not tiling in s. Notice also that x_0 is irredundant.

Observe that the set of motifs that are primitive (according to Definition 11) and not tiling contains at least all irredundant motifs x such that x occurs as a suffix of the input string. Indeed, in such cases, no motif different from x itself or occurring in it can cover the rightmost occurrence of x.

Definition 12 ($\mathcal{B}_P$) *Given a string s and a quorum q, the* basis of primitive motifs *$\mathcal{B}_P$ is the set of all primitive motifs for s and q according to Definition 11. The size of $\mathcal{B}_P$ is the number of motifs in $\mathcal{B}_P$.*

[1] In [12], page 331, $x = \texttt{AAAA}\circ\circ\texttt{AAAA}$ is listed among the primitive motifs with $q = 2$ of $\texttt{AAAAAXAAAAA}$ while, according to Definition 11, it is covered, for instance, by $\texttt{AAAA}\circ\circ\texttt{AAA}$ and $\texttt{AAAA}\circ\circ\texttt{AA}$ that occur at the same positions as x.

Example (continued) The basis $\mathcal{B}_P$ of the string $s =$ `CACACATACATAC` has size 3 and contains the motifs $x_0 =$ `AC` with $\mathcal{L}_{x_0} = \{1, 3, 7, 11\}$, $x_1 =$ `CA∘ACATAC` with $\mathcal{L}_{x_1} = \{0, 4\}$, and $x_2 =$ `CACA∘A∘A∘A` with $\mathcal{L}_{x_2} = \{0, 2\}$.

In the examples above we have shown an instance for which $\mathcal{B}_T \subsetneq \mathcal{B}_P \subsetneq \mathcal{B}_I$. In the Section 3 we will prove that these proper inclusions are valid in general.

2.5 Properties shared by all bases

In all definitions of a basis, a maximal motif x not in the basis is discarded by other motifs $y_1, \ldots, y_k$ distinct from x, each one occurring stricly less than $|\mathcal{L}_x|$ times. As a consequence, if x is a maximal motif that occurs exactly q times, then there exist no set of maximal motifs distinct from x that can cover or tile it because they should be motifs with strictly less than q occurrences and this would contradict tha fact that they are maximal. All maximal motifs occurring exactly q times belong therefore to all bases.

When, instead, a motif x is covered or tiles by others, say $y_1, \ldots, y_k$, then we must have by definition that $\mathcal{L}_{y_i}$ is a proper subset of $\mathcal{L}_x$ for $1 \leq i \leq k$ if x is covered, and $\mathcal{L}_{y_i}$ is a proper subset of $(\mathcal{L}_x - d_i)$ if x is tiled. Observe that in both cases, we have that x occurs in each one of the y_i's; this happens at position 0 of each y_i if x is redundant or non primitive, and at position d_i if it is tiled. In fact, a motif x is only tiled or covered by motifs that are more specific than it, and thus x occurs in each one of them.

Finally, in all definitions of a basis, we have that if the quorum is set to $q = 1$, then the input string itself is a maximal motif and it is the only maximal motif occurring exactly q times. In the case of tiling motifs, this single motif coincides also with the basis.

As we mentioned in Section 2.2, it is shown in [9, 10] that the set of irredundant motifs is enough to generate all maximal motifs. Basically, the reason is that the operation of eliminating a redundant motif can be easily reversed by means of the $\oplus$ operator. Indeed, the generating operator maps two or more patterns whose occurrences overlap into a pattern that is less specific than any of the initial patterns and whose occurrences is the union of all their occurrences.

It is intuitive to see that the $\oplus$ operator can be extended to allow a shift between the two input words in order to reverse also the operation of eliminating a tiled motif.

Definition 13 ($\oplus_h$) *Given h and $x, y \in (\Sigma \cup \circ)^*$ we define $u = x \oplus_h y$ by $u[i] = x[i+h] \oplus y[i]$ for all integers i between 0 and $|x| - h$.*

In practice, this is the same as performing the operation $\oplus$ between x, and y shifted by h positions to the right.

Example (continued) Let us consider the usual input string and the motif x_0 which is tiled by x_1 and x_2 as $\mathcal{L}_{x_0} = (\mathcal{L}_{x_1} + 7) \cup (\mathcal{L}_{x_2} + 1)$. We have that $x_1 \oplus_6 x_2 =$ `CA∘ACATAC` $\oplus_6$ `CACA∘A∘A∘A` $=$ `TAC` $\oplus$ `CACA∘A∘A∘AAC` $= x_0$. The shift $h = 6$ was chosen because $6 = 7 - 1$ where 7 and 1 are the respective shifts applied to $\mathcal{L}_{x_1}$ and to $\mathcal{L}_{x_2}$ in order to obtain $\mathcal{L}_{x_0}$ and thus to tile x_0.

In [10] an output sensitive algorithm is described which generates all maximal motifs starting from the basis of irredundant motifs (see Section 2.2). A similar approach applied to the basis of tiling motifs that replaces the $\oplus$ operator with its shifted version for all shifts would result in a method that generates all maximal motifs starting from such the basis. Making it also output sensitive is a less trivial extension because applying the $\oplus$ operator with different shifts to different pairs of tiling motifs may result in the same maximal (tiled) motif.

In either case however, we believe that actual applications based on the notion of a basis should not lead to a blind generation of all maximal motifs. Indeed, this would go against the motivation behind the whole idea of a basis of motifs. The generation of useful motifs should be driven by the specific application and should be applied on suitable subsets of the basis only. What is important is that, in theory, both bases are enough powerful to generate all motifs.

3 Differences among the bases

A remarkable difference between the bases is that the basis of tiling motifs $\mathcal{B}_T$ is symmetric. That is, each tiling motif in the basis for the reverse string $\widetilde{s}$ is the reverse of a tiling motif in the basis of s. This follows directly from the definition of tiling motifs and from the fact that maximality is also a symmetric notion. Symmetry is a desirable property for a basis. It does not hold for the bases $\mathcal{B}_I$ and $\mathcal{B}_P$ of, respectively, irredundant and primitive motifs. Intuitively, the reason is that a motif x may be eliminated by other motifs only if the starting positions of such motifs coincide with those of x, while the ending position may differ. When the string and the motifs (maximal or not) are reversed, the former ending positions become starting positions and they might no longer coincide.

Example (continued) Let us consider again the string $s = \mathtt{CACACATACATAC}$ and the irredundant (but not tiling) motif $x_6 = \mathtt{A}{\circ}\mathtt{A}$ for s. In the string $\widetilde{s} = \mathtt{CATACATACACAC}$ we have that the motif $\widetilde{x_6} = x_6$ is not irredundant because it is eliminated, for example, by $\mathtt{A}{\circ}\mathtt{AC} = \widetilde{x_3}$ and $\mathtt{A}{\circ}\mathtt{A}{\circ}\mathtt{A}{\circ}\mathtt{ACAC} = \widetilde{x_2}$. Indeed, $\mathcal{L}_{\widetilde{x_6}} = \{1, 3, 5, 7, 9\} = \{1, 5, 7, 9\} \cup \{1, 3\} = \mathcal{L}_{\widetilde{x_3}} \cup \mathcal{L}_{\widetilde{x_2}}$. Notice that motif x_3 is also maximal in s as well as its reverse in $\widetilde{s}$ (the notion of maximality is symmetric), but x_3 is redundant in s while its reverse is irredundant in $\widetilde{s}$. Similarly, motif $\widetilde{x_0} = \mathtt{CA}$ is not primitive in $\widetilde{s}$, while x_0 is primitive in s as shown in Section 2.4. Indeed, in $\widetilde{s}$ we have that $\widetilde{x_0}$ with $\mathcal{L}_{\widetilde{x_0}} = \{0, 4, 8, 10\}$ is covered, for example, by the two motifs $\mathtt{CAC}$ with $\mathcal{L}_{CAC} = \{8, 10\}$ and $\mathtt{CAT}$ with $\mathcal{L}_{CAT} = \{0, 4\}$.

We now show some results concerning set inclusions among the three bases. Before that, we need the following definition.

Definition 14 (Maximal Extension) *Given a string s and a motif y, the* maximal extension *of y is the motif $\overline{y}$ obtained by, as much as possible, extending y and replacing its don't cares with letters, so long as no occurrence is lost.*

Observe that for any y, the extension $\overline{y}$ is unique and that we always have that $\overline{y}$ is a maximal motif with that $\mathcal{L}_{\overline{y}} + d = \mathcal{L}_y$ for some $d \geq 0$.

Example (continued) In string $s = \texttt{CACACATACATAC}$, the maximal extension of the motif $y = \texttt{TAC}$ (with $\mathcal{L}_y = \{6, 10\}$) is $\overline{y} = \texttt{CA}\circ\texttt{ACATAC}$ (with $\mathcal{L}_{\overline{y}} = \{0, 4\}$).

We are now ready to prove the following results.

Lemma 1 *For any input string s and for any quorum q, we have that $\mathcal{B}_T \subseteq \mathcal{B}_P$. Moreover, there exist instances for which $\mathcal{B}_T \subsetneq \mathcal{B}_P$.*

Proof: In order to show that $\mathcal{B}_T \subseteq \mathcal{B}_P$, it is sufficient to show that for any $x \notin \mathcal{B}_P$ we have that $x \notin \mathcal{B}_T$. Let $x \notin \mathcal{B}_P$. This means that there exist k motifs $y_1, y_2, \ldots, y_k$ with $k > 1$ and $y_i \neq x$ such that $\mathcal{L}_x = \cup_{i=1}^{k} \mathcal{L}_{y_i}$. They are not necessarily maximal, hence let us consider $\overline{y}_1, \overline{y}_2, \ldots, \overline{y}_k$ that are, respectively, the *maximal extensions* of $y_1, y_2, \ldots, y_k$. The $\overline{y}_i$'s are maximal and none of them is equal to x because none of the y_i's occurred in x. Moreover, for each $1 \leq i \leq k$ we have that $\mathcal{L}_{\overline{y}_i} + d_i = \mathcal{L}_{y_i}$ for some $d_i \geq 0$, and thus $\mathcal{L}_x = \cup_{i=1}^{k} \mathcal{L}_{y_i} = \cup_{i=1}^{k} (\mathcal{L}_{\overline{y}_i} + d_i)$, meaning that the $\overline{y}_i$'s tile x. This proves that $x \notin \mathcal{B}_T$.

Finally, the string $s = \texttt{CACACATACATAC}$ of the previous examples is such that $\mathcal{B}_P \setminus \mathcal{B}_T = \{x_0\} \neq \emptyset$. ⋈

The following result is proved in [12].

Lemma 2 *For any input string s and for any quorum q, we have that $\mathcal{B}_P \subseteq \mathcal{B}_I$. Moreover, there are instances for which $\mathcal{B}_P \subsetneq \mathcal{B}_I$.*

Proof: As for the previous lemma, it is enough to show that for each $x \notin \mathcal{B}_I$ we have that $x \notin \mathcal{B}_P$. To this purpose, it is sufficient to show that the y_i's that cover x according to Definition 6 are maximal motifs and thus, in particular, they are motifs that cover x according to Definition 11 as well.

Finally, in the string s of the proof of Lemma 1, we have $\mathcal{B}_I \backslash \mathcal{B}_P = \{x_4, x_5, x_6, x_7\} \neq \emptyset$. ⋈

Summing up, we proved the following general result.

Theorem 1 *In general, $\mathcal{B}_T \subseteq \mathcal{B}_P \subseteq \mathcal{B}_I$; there are strings for which $\mathcal{B}_T \subsetneq \mathcal{B}_P \subsetneq \mathcal{B}_I$.*

The difference in the asymptotic sizes of the bases with respect to the length of the input string is an important issue that is addressed in Section 4 for $q = 2$ and in Section 5.2 for $q > 2$.

4 On the size of the bases if $q = 2$ and their construction

In this section we show that the different definitions for the bases lead to a hierarchy of space bounds, which enforces the condition $\mathcal{B}_T \subsetneq \mathcal{B}_P \subsetneq \mathcal{B}_I$ stated in Theorem 1, for an input string of length n and quorum $q = 2$. We demonstrate in Section 4.1 that the size of $\mathcal{B}_T$ is $O(n)$. In Section 4.2, we show that the size of $\mathcal{B}_P$ is between $\Omega(n)$ and $O(n^2)$, while we show in Section 4.3 that the size of $\mathcal{B}_I$ is $\Omega(n^2)$.

4.1 Tiling motifs

We define in this section a set of strings that are obtained by applying the $\oplus$ operation to the input string and all its possible shifts. We also give a characterization of tiling motifs in terms of these strings that leads to a linear upper bound for the number of tiling motifs.

Definition 15 (*Merge*) *Let* $Merge_k = s \oplus_k s$ *for* $1 \leq k \leq |s| - 1$.

In other words, given a string $s \in \Sigma^*$ (recall that strings are padded to the left and right with an infinite number of don't cares), $Merge_k$ is the result of the application of the $\oplus$ operation to the string s with s itself shifted k positions to the right.

Example (continued) Let us consider again the string $s =$ `CACACATACATAC`. The non empty merges are $Merge_2 =$ `CACA∘A∘A∘A`, $Merge_4 =$ `CA∘ACATAC`, $Merge_6 =$ `ACA∘A`, $Merge_8 =$ `CA∘AC`, and $Merge_{10} =$ `AC`.

We can observe that if $q = 2$ then each motif y occurs in at least one merge. In fact, it is enough to consider $Merge_{|j-i|}$ for $i, j \in \mathcal{L}_y$ with $i \neq j$. More precisely, each pattern y occurs $\binom{\mathcal{L}_y}{2}$ times in the merges, possibly having multiple occurrences in some of them. In fact, y has an occurrence in $Merge_{|j-i|}$ for each pair $i, j \in \mathcal{L}_y$ with $i \neq j$. Moreover, in [13, 14], the following lemmas are proved (and thus the proof is omitted here).

Lemma 3 *Any* $Merge_k$ *of length at least one is a maximal motif.*

Lemma 4 *Each tiling motif of a string* s *with* $q = 2$ *is equal to* $Merge_k$ *for some integer* k *between* 1 *and* $n - 1$.

As a direct consequence of Lemmas 3 and 4, we have the following two results.

Theorem 2 *Given a string* s *and* $q = 2$, $\mathcal{B}_T \subseteq \{Merge_k \mid 1 \leq k \leq n-1\}$.

Corollary 1 (Linearity of $\mathcal{B}_T$) *The number of elements in* $\mathcal{B}_T$ *of a string* s *of length* n *is at most* $n - 1$.

Example (continued) Among the merges listed in the previous example, we have that $Merge_1$ and $Merge_2$ coincide with, respectively, x_2 and x_1, which are precisely the only tiling motifs of s. The other merges are all maximal but tiled.

In [15], an $O(n^2 \log n \log |\Sigma|)$ time complexity (and linear space) algorithm is given for the computation of the basis of tiling motifs. The algorithm is strongly based on the property of tiling motifs stated in Theorem 2. The algorithm is sketched as follows.

- The multiset $\mathcal{M}'$ of the $n - 1$ merges is computed. For each $Merge_k$, compute the occurrences list $\mathcal{T}_k = \{\delta, \delta + k\}$ where $s[\delta] = s[\delta + k]$ is the first solid character of $Merge_k$ (done in $O(n^2)$ time).

- The multiset $\mathcal{M}'$ is transformed into the set $\mathcal{M}$ by removing duplicates and possibly joining the lists $\mathcal{T}$ (takes $O(n^2 \log |\Sigma|)$ time).
- The complete occurrence lists $\mathcal{L}$ of the merges are computed and the merges such that $\mathcal{L} \neq \mathcal{T}$ are discarded[2] (done in $O(n^2 \log n \log |\Sigma|)$ time).
- The remaining merges have a total number of occurrences that is linear in $|s|$, which allows to efficiently discard all remaining tiled motifs, if any (in $O(n^2)$ time).

More details about the algorithm can be found in [13, 14, 15].

4.2 Primitive motifs

The analog of Theorem 2 does not hold for $\mathcal{B}_P$ as the following counter-example shows.

Example Let us consider the string $s' = \texttt{CACACATACACAC}$, quorum $q = 2$, and motif $x'_0 = \texttt{AC}$ with $\mathcal{L}_{x'_0} = \{1, 3, 7, 9, 11\}$. We have that x'_0 is maximal because a right extension would cause the loss of the rightmost occurrence, a left extension with a C would cause the loss of the occurrence at position 7, and any longer left extension would exclude the leftmost occurrence. Moreover, x'_0 is primitive because no motif different from x'_0 itself can cover the occurrence at position 11. The merges of s' are $\mathit{Merge}_2 = \texttt{CACA}\circ\texttt{A}\circ\texttt{ACAC}$, $\mathit{Merge}_4 = \texttt{CA}\circ\texttt{ACA}\circ\texttt{AC}$, $\mathit{Merge}_6 = \texttt{ACACA}$, $\mathit{Merge}_8 = \texttt{CACAC}$, $\mathit{Merge}_{10} = \texttt{CAC}$, and $\mathit{Merge}_{12} = \texttt{C}$. The primitive motif x'_0 is not equal to any of the merges, although it is the suffix of several of them.

In order to understand the differences illustrated in the example, we give a characterization of the motifs in $\mathcal{B}_P \setminus \mathcal{B}_T$, which allows to obtain an upper bound on the size of such set, and hence on the size of $\mathcal{B}_P$ for $q = 2$.

Lemma 5 $(\mathcal{B}_P \setminus \mathcal{B}_T) \subset \{x' \mid x' \textit{ is a suffix of } x \in \mathcal{B}_T\}$.

Proof: Let $x \in (\mathcal{B}_P \setminus \mathcal{B}_T)$. Since x is tiled, there must exist tiling motifs $y_1, y_2, \ldots, y_k$ and integers $d_1, d_2, \ldots, d_k$ such that $\mathcal{L}_x = \cup_{i=1}^{k} (\mathcal{L}_{y_i} + d_i)$ and $y_i \neq x$ for each $1 \leq i \leq k$. Let us now consider, for each $1 \leq i \leq k$, the pattern y'_i that is the suffix of y_i starting at position d_i. The patterns $y'_1, y'_2, \ldots, y'_k$ have the following properties:

- They start with a solid character. The reason for this is that the y_i's tile x with shifts d_i's. Their positions $y_i[d_i]$ thus coincide with the first letter of x which is the same in all occurrences. Since the y_i's are maximal, we have that $y_i[d_i] = x[0] \in \Sigma$.
- They occur at least q times because they are suffixes of motifs.
- We have that $\mathcal{L}_x = \cup_{i=1}^{k} \mathcal{L}_{y'_i}$ because $\mathcal{L}_{y'_i} = \mathcal{L}_{y_i} + d_i$. In fact, we have that $(\mathcal{L}_{y_i} + d_i) \subseteq \mathcal{L}_{y'_i}$ because y'_i occurs at position d_i of y_i and thus that $\mathcal{L}_x \subseteq \cup_{i=1}^{k} \mathcal{L}_{y'_i}$. Moreover, we have that x occurs in each y'_i and thus that $\cup_{i=1}^{k} \mathcal{L}_{y'_i} \subseteq \mathcal{L}_x$.

[2] Correctness of this step is guaranteed by a result proved in [13] stating that for all tiling motifs the lists $\mathcal{L}$ and $\mathcal{T}$ must coincide.

Hence, if $y'_1, y'_2, \ldots, y'_k$ do not cover x (which is primitive), it must be because at least one of them is equal to x, which means that x is a suffix of the tiling motif y_i. $\bowtie$

It is worth observing that the primitive motif x'_0 of the last example meets the characterization of Lemma 5 because it is a suffix of $\mathit{Merge}_2, \mathit{Merge}_4, \mathit{Merge}_6, \mathit{Merge}_8$ which are tiling motifs for s'. We can now state the following result concerning the size of $\mathcal{B}_P$ if $q = 2$.

Theorem 3 *The number of elements in $\mathcal{B}_P$ of a string s of length n with $q = 2$ is $O(n^2)$.*

Proof: This is a direct consequence of the fact that $|\mathcal{B}_T| = O(n)$ and that $|\mathcal{B}_P \setminus \mathcal{B}_T| = O(n^2)$ since there are at most $O(n^2)$ suffixes of motifs in $\mathcal{B}_T$ if $q = 2$. $\bowtie$

Observe that the quadratic upper bound of Theorem 3 may not be tight and thus the size of $\mathcal{B}_P$ can still be linear if $q = 2$, although it is not a subset of the set of the merges.

Finally, an $O(|\Sigma| n^2 \log^2 n \log\log n)$ time complexity algorithm for the computation of the set of primitive motifs if $q = 2$ is described in [12]. The idea of the algorithm is also to first compute a superset of the basis, and then to check for the primitive property. However, it is not clear for which one of the two definitions (primitive or tiling motifs) given in [12] the algorithm is designed. Furthermore, the paper assumes that there is at most a linear number of primitive motifs for $q = 2$.

4.3 Irredundant motifs

In this section, we show a quadratic lower bound for the number of irredundant motifs. We can already observe that the set of irredundant motifs is not a subset of the merges, even though each irredundant motif necessarily occurs in at least one merge. Indeed, any irredundant motif x must occur in Merge_k for $k = j - i$ with $i, j \in \mathcal{L}_x$ and $i < j$.

Example (continued) In the string $s = \texttt{CACACATACATAC}$, the merges are $\mathit{Merge}_2 = \texttt{CACA}\circ\texttt{A}\circ\texttt{A}\circ\texttt{A}$, $\mathit{Merge}_4 = \texttt{CA}\circ\texttt{ACATAC}$, $\mathit{Merge}_6 = \texttt{ACA}\circ\texttt{A}$, $\mathit{Merge}_8 = \texttt{CA}\circ\texttt{AC}$, and $\mathit{Merge}_{10} = \texttt{AC}$. The irredundant motifs are $x_0 = \texttt{AC} = \mathit{Merge}_{10}$, $x_1 = \texttt{CA}\circ\texttt{ACATAC} = \mathit{Merge}_4$, $x_2 = \texttt{CACA}\circ\texttt{A}\circ\texttt{A}\circ\texttt{A} = \mathit{Merge}_2$, $x_4 = \texttt{CA}\circ\texttt{AC} = \mathit{Merge}_8$, $x_5 = \texttt{ACA}\circ\texttt{A} = \mathit{Merge}_6$, $x_6 = \texttt{A}\circ\texttt{A}$ which occurs in Merge_2, Merge_4, Merge_6, and Merge_8 but is not equal to any of them, and finally $x_7 = \texttt{A}\circ\texttt{A}\circ\texttt{A}$ which occurs in (but is different from) Merge_2.

Hence, the $n - 1$ upper bound cannot hold for irredundant motifs. We now show that no linear upper bound holds either because there exists an infinite family of strings, introduced in [14], for which there are $\Omega(n^2)$ irredundant motifs in the basis for quorum $q = 2$, where n is the length of the input string. Such strings are denoted by s_k. Each one of them (for any positive integer value of k) is an extension of $t_k = \texttt{A}^k\texttt{T}\texttt{A}^k$, where $\texttt{A}^k$ denotes the letter $\texttt{A}$ repeated k times. String t_k has an exponential number of maximal motifs, including those having the form $\texttt{A}\{\texttt{A}, \circ\}^{k-2}\texttt{A}$ with exactly two don't cares. The reason is that each such motif, let us call it x, occurs four times in t_k: two occurrences of x match the first and the last k letters in t_k while each distinct don't care in x matching the letter $\texttt{T}$ in t_k contributes to one of the two remaining occurrences. Extending x or replacing a don't care with a solid character reduces the number of these occurrences, so x is maximal. However, each motif x is covered in t_k and hence it is redundant. String s_k is obtained by prefixing t_k with $O(|t_k|)$ symbols so that each motif x becomes irredundant in s_k. Since there are $\Omega(k^2)$ of them, and $n = |s_k| = \Theta(|t_k|) = \Theta(k)$, this leads to the quadratic lower bound for irredundant motifs.

In order to define the string s_k on the alphabet $\Sigma = \{\mathtt{A}, \mathtt{T}, \mathtt{u}, \mathtt{v}, \mathtt{w}, \mathtt{x}, \mathtt{y}, \mathtt{z}, \mathtt{a}_1, \mathtt{a}_2, \ldots, \mathtt{a}_{k-2}\}$, we introduce a few notations. Let $\widetilde{u}$ denote the reversal of u, and let ev_k, od_k, u_k, v_k be the strings defined as follows:

$$\begin{array}{rrcl@{\qquad}rrcl}
\text{if } k \text{ is even :} & ev_k & = & \mathtt{a}_2\mathtt{a}_4\cdots\mathtt{a}_{k-2}, & \text{if } k \text{ is odd :} & ev_k & = & \mathtt{a}_2\mathtt{a}_4\cdots\mathtt{a}_{k-3},\\
& od_k & = & \mathtt{a}_1\mathtt{a}_3\cdots\mathtt{a}_{k-3}, & & od_k & = & \mathtt{a}_1\mathtt{a}_3\cdots\mathtt{a}_{k-2},\\
& u_k & = & ev_k\,\mathtt{u}\,\widetilde{ev_k}\,\mathtt{vw}\,ev_k, & & u_k & = & ev_k\,\mathtt{uv}\,\widetilde{ev_k}\,\mathtt{wx}\,ev_k,\\
& v_k & = & od_k\,\mathtt{xy}\,\widetilde{od_k}\,\mathtt{z}\,od_k, & & v_k & = & od_k\,\mathtt{y}\,\widetilde{od_k}\,\mathtt{z}\,od_k.
\end{array}$$

The strings s_k are then defined by $s_k = u_k v_k t_k$ for $k \geq 5$. For example, denoting letter $\mathtt{a_i}$ with the number $\mathtt{i}$, we have that $s_7 = \mathtt{24uv42wx24135y531z135AAAAAAATAAAAAAA}$.

Whatever the parity of k is, the string s_k has size $n = \Theta(k)$. Moreover, in [14] the following result is proved.

Proposition 1 *Each motif of the form* $\mathtt{A}\{\mathtt{A}, \circ\}^{k-2}\mathtt{A}$ *with exactly two don't cares is irredundant in* s_k*.*

By Proposition 1, the number of irredundant motifs in s_k is at least $\binom{k-2}{2} = \Omega(k^2)$, that is the number of choices of two positions in $\{\mathtt{A}, \circ\}^{k-2}$. Since we observed that $n = O(k)$, then we have that the basis of irredundant motifs for string s_k contains $\Omega(n^2)$ irredundant motifs. All the details of the proof of this lower bound can be found in [14].

The construction of the basis for the irredundant motifs given in [1] is incremental and requires $O(n^3)$ time for an input string s of length n. Let $\mathcal{B}^i$ denote the basis of irredundant motifs for the suffix $s_i = s[i]s[i+1]\cdots s[n-1]$ of the input string. The base step is $\mathcal{B}^{n-1}$, which is simple to build. The inductive step computes $\mathcal{B}^i$ having already computed $\mathcal{B}^{i+1}$. The final outcome is then $\mathcal{B}^0$, which is the basis of irredundant motifs for s.

In order to illustrate the inductive step, we may figure out an $n \times n$ matrix M, whose rows and columns are numbered from 0 to $n-1$. Row i and column i are associated with symbol $s[i]$, so that $M[i, j] = s[i] \oplus s[j]$. Only the entries for which $i < j$ are filled, so that M is an upper triangular matrix.

The diagonals of M restricted to its rows and columns numbered from i to $n-1$ contribute to the construction of $\mathcal{B}^i$. In particular, to obtain $\mathcal{B}^i$ from $\mathcal{B}^{i+1}$, two steps are run adding row i and column i to M:

- FIND: Build $\hat{\mathcal{B}}$, which is the set of the motifs in $\mathcal{B}^{i+1}$ that occur also in position i, plus those occurring in i but not contained in $\mathcal{B}^{i+1}$.

- DISCARD: Eliminate the redundant motifs in $\hat{\mathcal{B}} \cup \mathcal{B}^{i+1}$, thus obtaining $\mathcal{B}^i$.

Several properties are exhibited in [1] to show that the two steps take each $O(n^2)$ time, for a total of $O(n^3)$ time. We refer the reader to that paper for more details.

5 Increasing the quorum

We now indicate some properties related to the computation of a basis for a string s when the quorum q is higher than 2. We start by observing some properties of maximal motifs related to the quorum.

Definition 16 (Max_q) *Given a string s, let Max_q be the set of maximal motifs for a quorum q (with $q \geq 2$).*

We start by proving the following result.

Lemma 6 *Given a string s, we have that $Max_{q'} \subseteq Max_q$ for all $2 \leq q < q'$.*

Proof: Let $x \in Max_{q'}$. Suppose $x \notin Max_q$. This implies there exists $y \in Max_q$ with $\mathcal{L}_y = \mathcal{L}_x + d$ and y does not occur in x. Therefore, x occurs in y (consequently, $d \leq 0$). Motif y has thus the same number of occurrences as x (this number is at least q') and is more specific than it, which contradicts the hypothesis that $x \in Max_{q'}$. $\bowtie$

Lemma 6 applied to increasing values of the quorum suggests that for each string $s \in \Sigma^n$, there exists q_0 with $2 \leq q_o \leq n$ such that, for $q > q_0$, we have $Max_q = \emptyset$, and thus also all the bases eventually collapse to the empty set when the quorum reaches the threshold q_0. The value of q_0 is indeed the number of occurrences of the most frequent symbol appearing in the input string. In general, it can be as large as n, as is the case for $s = A^n$ where $\{A\} = Max_n = \mathcal{B}_T = \mathcal{B}_P = \mathcal{B}_I$ with $q = n$. Interesting values of q are those for which this is not yet the case and in the next section we investigate some properties of the basis of tiling motifs with quorum $q > 2$ with respect to the bases with smaller quorums.

5.1 Generating $\mathcal{B}_{q+1}$ from $\mathcal{B}_q$ for tiling motifs

In this section, we show some properties of the basis of tiling motifs for growing values of the quorum. Related results have been independently observed also in [12] for primitive motifs. In what follows, the results are proved for tiling motifs, thus in this section $\mathcal{B}$ will mean $\mathcal{B}_T$ omitting the subscript $_T$, which we replace with the value of the quorum q. When appropriate, we shall mention also whether the property holds for the other bases.

Definition 17 ($\mathcal{B}_q$) *Given a string s and quorum q, let $\mathcal{B}_q \subseteq Max_q$ be the set of tiling motifs with quorum q (with $q \geq 2$).*

We start by defining an useful notation.

Definition 18 ($\mathcal{B}_q^=, \mathcal{B}_q^>$) *Let $\mathcal{B}_q^= = \{x \in \mathcal{B}_q \mid |\mathcal{L}_x| = q\} \subseteq \mathcal{B}_q$ be the set of motifs in $\mathcal{B}_q$ that occur exactly q times. In a similar way, we denote by $\mathcal{B}_q^> = \{x \in \mathcal{B}_q \mid |\mathcal{L}_x| > q\} \subseteq \mathcal{B}_q$ the set of motifs in $\mathcal{B}_q$ that occur strictly more than q times. Clearly $\mathcal{B}_q^= \cup \mathcal{B}_q^> = \mathcal{B}_q$ and $\mathcal{B}_q^= \cap \mathcal{B}_q^> = \emptyset$.*

We now prove a collection of properties that result in a characterization of the motifs in $\mathcal{B}_{q+1}$ in terms of those in $\mathcal{B}_q$.

The following result also holds for primitive ([12]) and irredundant motifs (the proof can be extended in a straightforward way).

Lemma 7 *For all $q \geq 2$, any motif $x \in \mathcal{B}_q^{>}$ belongs also to $\mathcal{B}_{q'}$ for $q < q' \leq |\mathcal{L}_x|$.*

Proof: If x is in $\mathcal{B}_q^{>}$, then it is maximal and, since it occurs at least q' times for $q < q' \leq |\mathcal{L}_x|$, then $x \in Max_{q'}$. Suppose by absurd, x were tiled for such quorum $q' \leq |\mathcal{L}_x|$. Let y_1, y_2, $\ldots$, y_k be the maximal motifs which tile x, that is, such that $\mathcal{L}_x = \cup_{i=1}^{k}(\mathcal{L}_{y_i} + d_i)$ for integers d_1, d_2, $\ldots$, d_k. By Lemma 6, if y_1, y_2, $\ldots$, $y_k \in Max_{q'}$, then y_1, y_2, $\ldots$, $y_k \in Max_q$ for $q < q'$. This means that x would be tiled by motifs in Max_q, and thus that x is not in $\mathcal{B}_q$, a contradiction. $\bowtie$

It is obvious that a motif x in $\mathcal{B}_q^{=}$ cannot belong to $\mathcal{B}_{q'}$ for any $q' > q$ because it does not satisfy the quorum. On the other hand, the reverse of Lemma 7 is not true: a motif x in $\mathcal{B}_{q'}$ may not belong to $\mathcal{B}_q$ for $q < q'$ because there can be motifs in $Max_q - Max_{q'}$ that tile it. Therefore, although the basis is always a subset of the set of maximal motifs, the inclusion relation proved in Lemma 6 for Max_q is not inherited by $\mathcal{B}_q$. In fact, in general, the set $\mathcal{B}_q$ is not a subset nor a superset of $\mathcal{B}_{q-1}$. This holds for all the bases as shown by the following example.

Example Let us consider the string $s =$ ACAAATAAAAA. With quorum 2, basis $\mathcal{B}_2$ contains A∘∘AA∘AA, A∘A∘A∘AAA, AA∘∘AAAA, and A∘AAA∘A. Basis $\mathcal{B}_3$ includes A∘∘∘A∘AA, A∘∘AA∘A, A∘AAA. We see that $\mathcal{B}_2$ and $\mathcal{B}_3$ do not even intersect. In fact, all motifs in $\mathcal{B}_2$ occur exactly twice and thus cannot belong to Max_3, and therefore to $\mathcal{B}_3$. On the other hand, all motifs in $\mathcal{B}_3$ certainly belong to Max_3 and hence to Max_2, but they are not tiling because they are tiled by motifs that belong to $Max_2 \setminus Max_3$. For example, AA∘∘A with $\mathcal{L}_{\mathtt{AA\circ\circ A}} = \{3, 4, 7\}$ is in $\mathcal{B}_3$, but it is not in $\mathcal{B}_2$ because it is tiled by AA∘∘AAAA (with $\mathcal{L}_{\mathtt{AA\circ\circ AAAA}} = \{3, 4\}$) and A∘∘AA∘AA (with $\mathcal{L}_{\mathtt{A\circ\circ AA\circ AA}} = \{1, 4\}$) shifted by three positions. For the same string, the bases of primitive and irredundant motifs for $q = 2$ contain the tiling motifs plus some others (the primitives non tiling for $q = 2$ are AA, AAA, and A∘AAA, and the irredundant non primitive are A∘∘∘AAA, A∘AA, and AA∘∘A). For both bases, the motifs that are also tiling must be discarded when q goes from 2 to 3 because they lose the quorum. Moreover, A∘∘AA∘A is primitive and irredundant (because it is tiling) when $q = 3$ but neither of them when $q = 2$.

We recall that $u = x \oplus_h y$ is such that $u[i] = x[i + h] \oplus_h y[i]$ for all integers i between 0 and $|x| - h$, that is the operation applies to y and the i^{th} suffix of x, which we also denote by $x[i \mathinner{..} |x| - 1]$. The operation $\oplus$ can be naturally extended to more than two strings by placing a don't care at a position where at least two of the strings have different characters. Notice that such operation $\oplus$ is clearly associative (and commutative), while $\oplus_h$ in general is not. For instance, we have that $((\mathtt{ACDEG{\circ}H} \oplus_2 \mathtt{DAGMH}) \oplus_2 \mathtt{GM}) = \mathtt{G}$ is different from $(\mathtt{ACDEG{\circ}H} \oplus_2 (\mathtt{DAGMH} \oplus_2 \mathtt{GM}))$ that results in the empty string. Therefore, when we apply the shifted $\oplus$ to more than two strings, we instead use a notation that applies the shift to the string (rather than to the operator) and we use the unshifted operator $\oplus$ notation. Hence, we use the more intuitive notation $x[i\,..\,|x|-1] \oplus y[j\,..\,|y|-1] \oplus w$ to indicate that the operator $\oplus$ is applied to x, y, and w shifted in such a way that position i of x is aligned with position j of y and position 0 of w, although this can be written in a more compact way with an adequate choice of the shift applied to the operator.

Lemma 8 *Let y, z be two distinct elements of $\mathcal{B}_q$ for $q \geq 2$ padded to the left and to the right with an infinite number of don't cares. Let $x = (y \oplus_i z) \in \Sigma \cup \Sigma(\Sigma \cup \{\circ\})^*\Sigma$ for i an integer between 0 and $|y|-1$. If x is not empty, then $|\mathcal{L}_x| > q$. Moreover, motif x belongs to $Max_{q'}$ for $q < q' \leq |\mathcal{L}_x|$.*

Proof: Clearly $|\mathcal{L}_x| \geq q$. Suppose $|\mathcal{L}_x| = q$. Since $\mathcal{L}_x = (\mathcal{L}_y + i) \cup \mathcal{L}_z$, then $|\mathcal{L}_x| = q = |\mathcal{L}_y + i| = |\mathcal{L}_z|$ and y and z would not be distinct, which is a contradiction. We now prove that $x \in Max_{q'}$. Suppose again by absurd that $x \notin Max_{q'}$ for $q < q' \leq |\mathcal{L}_x|$. There exists $w \in Max_{q'}$ with $\mathcal{L}_w = \mathcal{L}_x + d$ such that w does not occur in x. Therefore, x occurs in w. Let j and k be the positions of the occurrences of x in $y[i\,..\,|y|-1]$ and z respectively. Let $y' = w[d\,..\,|w|-1] \oplus y[i+j\,..\,|y|-1]$ and $z' = w[d\,..\,|w|-1] \oplus z[k\,..\,|z|-1]$. We have that $y[0]\cdots y[i+j-1]y'$ and $z[0]\cdots z[k-1]z'$ are more specific than y and z respectively, contradicting the hypothesis that $y, z \in \mathcal{B}_q$. ⋈

In other words, we have shown that a pair of motifs in $y, z \in \mathcal{B}_q$ can generate a third motif x having more than q occurrences. Such a motif is not in $\mathcal{B}_q$ because it is tiled by y and z, but it is a candidate to belong to the basis for quorum higher than q because it is maximal and it is possible that for a higher quorum, the set of maximal motifs no longer contains motifs that tile it. The next result investigates what happens to x in the case where $|\mathcal{L}_y|, |\mathcal{L}_z| \geq q+1$.

Lemma 9 *Let y, z be two distinct elements of $\mathcal{B}_q^>$ for $q \geq 2$. Let $x = y \oplus_i z \in \Sigma \cup \Sigma(\Sigma \cup \{\circ\})^*\Sigma$ for i an integer between 0 and $|y|-1$ such that x is not empty. We have that x belongs to $Max_{q+1} - \mathcal{B}_{q+1}$.*

Proof: The fact that $x \in Max_{q+1}$ is an immediate consequence of Lemma 7. Moreover, $x \notin \mathcal{B}_{q+1}$ because it is tiled by y and z, which also belong to Max_{q+1}. ⋈

A simple generalization of Lemma 9 suggets that a motif x generated by a pair of motifs y and z in $\mathcal{B}_q^>$ is a candidate for belonging to a basis only for a quorum stricly greater than $\min\{|\mathcal{L}_y|, |\mathcal{L}_z|\}$.

Lemma 10 *Let $y_1, \ldots, y_k$ (with $k > 2$) be three or more distinct elements of $\mathcal{B}_q$ for $q \geq 2$ padded to the left and to the right with an infinite number of don't cares. Let the string x be such that $x = (y_1[i_1\,..\,|y_1|-1] \oplus y_2[i_2\,..\,|y_2|-1] \oplus \cdots \oplus y_k) \in \Sigma \cup \Sigma(\Sigma \cup \{\circ\})^*\Sigma$ belong to Max_{q+1} for $i_1, \ldots, i_{k-1}$ integers such that $0 \leq i_j \leq |y_j|-1$ and $1 \leq j \leq k$. Then x does not belong to $\mathcal{B}_{q+1}$.*

Proof: We prove the results for the case of three strings, say y_1, y_2 and y_3, since the generalization to any number $k > 2$ of strings is straightforward. Let $r_1 = y_1[i_1\,..\,|y_1|-1] \oplus y_2[i_2\,..\,|y_2|-1]$ and $r_2 = y_2[i_2\,..\,|y_2|-1] \oplus y_3$. By Lemma 8, $r_1, r_2 \in Max_{q+1}$ and thus also $|\mathcal{L}_{r_1}|, |\mathcal{L}_{r_2}| > q$. Since $x = y_1[i_1\,..\,|y_1|-1] \oplus y_2[i_2\,..\,|y_2|-1] \oplus y_3 \in Max_{q+1}$ and $y_1[i_1\,..\,|y_1|-1] \oplus y_2[i_2\,..\,|y_2|-1] \oplus y_3 = r_1[i_1\,..\,|r_1|-1] \oplus r_2$, with $|\mathcal{L}_{r_1}|, |\mathcal{L}_{r_2}| \geq q+1$, maximal motifs r_1 and r_2 tile motif x. ⋈

Summing up, as we state in the following proposition, Lemmas 8, 9 and 10 tell us that two motifs in $\mathcal{B}_q$ can generate a motif in $\mathcal{B}_{q+1}$ only if one of the two has exactly q occurrences, and that more than two motifs in $\mathcal{B}_q$ cannot generate a motif in $\mathcal{B}_{q+1}$, even if they all have exactly q occurrences.

Proposition 2 *Let x belong to $\mathcal{B}_{q+1}$. We have that, if $x \notin \mathcal{B}_q$ then there exist $y, z \in \mathcal{B}_q$ with either $y \in \mathcal{B}_q^{=}$ or (inclusive) $z \in \mathcal{B}_q^{=}$, and an integer i between 0 and $|y| - 1$ such that $x = y \oplus_i z$.*

Proof: If $x \in \mathcal{B}_{q+1}$, then it must be that $x \in Max_{q+1}$ and thus $x \in Max_q$. Therefore, $s \in Max_q \setminus \mathcal{B}_q$, and thus there exists $y_1, \ldots, y_k \in Max_q$ that tile x. For Lemma 10, it must be $k = 2$. Finally, Lemma 9 tells us that y_1 or y_2 belong to $\mathcal{B}_q^{=}$, which is the thesis. $\bowtie$

A consequence of the results introduced in this section is that there are necessary and sufficient conditions on motifs in $\mathcal{B}_q$ that allow an incremental generation of the bases for higher quorum. This is actually what was done in [12] where such an incremental algorithm generating a basis with quorum q from a basis with quorum $q - 1$ and from Max_{q-1} is suggested. Nevertheless, this algorithm has an exponential time complexity. This follows immediately from the results shown in the next section.

5.2 An exponential lower bound for all bases

We show (Section 5.2.1) that *no* polynomial-time algorithm generating bases explicitly can exist for *any* arbitrary value of q in the worst case, for all the bases considered in this paper. The size of these bases can be proven to depend *exponentially* on $q \geq 2$, that is, we give a lower bound of $\binom{\frac{n-1}{2}-1}{q-1} = \Omega(\frac{1}{2^q}\binom{n-1}{q-1})$ for the size of any basis. In practice, this size has an exponential growth for increasing values of q up to $O(\log n)$, but larger values of q are theoretically possible in the worst case. Fixing $q = (n-1)/4 + 1$ in our lower bound, we get a size of $\Omega(2^{(n-1)/4})$ motifs in the bases. On the average, $q = O(\log_{|\Sigma|} n)$ using the fact that the expected number of simultaneous comparisons needed to find the first solid character of a merge is $O(|\Sigma|^{q-1})$, which must be less than n.

We show a further property for the basis of tiling motifs in Section 5.2.2, giving an upper bound of $\binom{n-1}{q-1}$ on its size with a simple proof. Since we can find an algorithm taking time proportional to the square of that size, we can conclude that a polynomial-time algorithm for finding the basis of tiling motifs exists in the worst case if and only if the quorum q satisfies either $q = O(1)$ or $q = n - O(1)$ (the latter condition is hardly meaningful in practice).

5.2.1 A lower bound of $\binom{\frac{n-1}{2}-1}{q-1}$ on the bases

We show the existence of a family of strings for which there are at least $\binom{\frac{n-1}{2}-1}{q-1}$ tiling motifs for a quorum q. Since a tiling motif is also primitive and irredundant, this gives a lower bound for all bases. For $q > 2$, this gives a lower bound of $\Omega\binom{\frac{n-1}{2}-1}{q-1} = \Omega(\frac{1}{2^q}\binom{n-1}{q-1})$ for the the size of all bases.

The strings are this time of the form $t_k = \mathtt{A}^k\mathtt{T}\mathtt{A}^k$ ($k \geq 5$), without the left extension used in the bound of Section 4.3. The proof proceeds by exhibiting $\binom{k-1}{q-1}$ motifs that are maximal and have each exactly q occurrences, from whence it follows immediately that they are tiling.

Indeed, the observation we made in Section 2.5 holds for any $q \geq 2$. Namely, all maximal motifs that occur exactly q times in a string are tiling. The following results are proved in [14].

Proposition 3 *For $2 \leq q \leq k$ and $1 \leq p \leq k - q + 1$, any motif $\mathtt{A}^p \circ \{\mathtt{A}, \circ\}^{k-p-1} \circ \mathtt{A}^p$ with exactly q don't cares is tiling (and thus primitive and irredundant) in t_k.*

Theorem 4 *String t_k has $\binom{\frac{n-1}{2}-1}{q-1} = \Omega(\frac{1}{2^q}\binom{n-1}{q-1})$ tiling (and thus primitive and irredundant) motifs, where $n = |t_k|$ and $k \geq 2$.*

5.2.2 An upper bound of $\binom{n-1}{q-1}$ tiling motifs

We now show that $\binom{n-1}{q-1}$ is an upper bound for the size of a basis of tiling motifs for a string s and quorum $q \geq 2$. Let us denote as before such a basis by $\mathcal{B}$. To prove the upper bound, we use again the notion of a merge except that it involves q strings. Let k denote now an *array* of $q-1$ positive values $k_1, \ldots, k_{q-1}$ with $1 \leq k_i < k_j \leq n-1$ for all $1 \leq i < j \leq q-1$. A merge is the (non empty) pattern that results from applying the operator $\oplus$ to the string s and to s itself $q-1$ times, each time shifted by k_i positions to the right for $1 \leq i \leq q-1$.

Definition 19 (*MMerge*) *Let s_k denote the string such that its jth character is $s_k[j] = s[j] \oplus s[j+k_1] \oplus \cdots \oplus s[j+k_{q-1}]$ for all integers j. MMerge$_k$ is the pattern obtained by removing all the leading and trailing don't cares in s_k (that is, appearing before the leftmost solid character and after the rightmost solid character).*

Lemmas 11–12 reported below extend Lemmas 3–4 to the case of $q > 2$ and their proofs are also simple extensions.

Lemma 11 *If MMerge$_k$ exists for quorum q, it must be a maximal motif.*

Lemma 12 *For each tiling motif x in the basis $\mathcal{B}$ with quorum q, there is at least one k for which MMerge$_k = x$.*

We now state the main property of tiling bases that follows from Lemma 11 and Lemma 12.

Theorem 5 *Given a string s of length n and a quorum $q \geq 2$, let $\mathcal{M}$ be the set of MMerge$_k$, for any of the $\binom{n-1}{q-1}$ possible choices of k for which MMerge$_k$ exists. The basis $\mathcal{B}$ of tiling motifs for s satisfies $\mathcal{B} \subseteq \mathcal{M}$, and therefore the size of $\mathcal{B}$ is at most $\binom{n-1}{q-1}$.*

The tiling motifs in our basis appear in s for a total of $q\binom{n-1}{q-1}$ times at most. A variation of the algorithm given in [15] gives a pseudo-polynomial time complexity of $O\left(q^2\binom{n-1}{q-1}^2\right)$ for computing $\mathcal{B}_q$.

In [12], the same upper bound is proved (independently from the proof of [14]). Namely, it is shown that there are at most C_{n-1}^{q-1} *primitive* motifs, but again here it is not clear whether the result refers to the definition of tiling motifs or of primitive ones. The same doubt applies to an algorithm that computes the basis of *primitive* motifs for quorum q from that with quorum $q-1$ in $O(C_{n-1}^{q-1}\,|\Sigma|\,n\log^2 n\log\log n)$ time. Adding the cost of computing the requested basis for $q-1$ (and thus for $q-2$ and so on until the quorum 2), this would also result in an algorithm computing a basis for any q having a cost that is quadratic in the possibly exponential size of the basis itself. Notice that Lemma 5 which characterizes the set $\mathcal{B}_P \setminus \mathcal{B}_T$ is valid for any quorum q, and hence we have that in general the size of $\mathcal{B}_P$ is in $O(|\mathcal{B}_T| + n|\mathcal{B}_T|) = O(n|\mathcal{B}_T|)$ but, as it was the case for $q = 2$, this bound is not necessarily tight.

Combining the upper bound shown in this section with the lower bound of Section 5.2.1, we obtain that there exists a polynomial time algorithm for finding the basis if and only if either $q = O(1)$ or $q = n - O(1)$.

6 Two Bases for Common Motifs

In the literature, the problem of finding motifs that are *common* to more than one string instead of repeated within a unique input string is often considered. For instance, a motif common to two or more biological sequences can correspond to a binding site shared by the sequences. The problem has also other text mining applications that we mentioned in Section 1, such as detection of plagiarism and frequent itemsets. In this section, we define two notions of basis for the set of motifs common to two input strings.

6.1 Weak Basis

Let $s \in \Sigma^n$ and $t \in \Sigma^m$ be the two input strings (we assume w.l.o.g. that $m \geq n$). As for the case of repeated motifs considered so far, common motifs are strings in the alphabet $\Sigma \cup \{\circ\}$ where Σ is the alphabet of the input strings and $\circ$ is the don't care. Such strings start and end with a solid character. Differently from the problem addressed so far, there is no other quorum requirement for common motifs than that they should occur in both input strings. Assuming the definitions of pattern, specificity, and occurrence of Section 2, we extend the definitions concerning motifs to common motifs and their occurrences in the following way.

Definition 20 (Common motif) *A pattern $x \in \Sigma \cup \Sigma(\Sigma \cup \{\circ\})^*\Sigma$ is a* common motif *for strings $s \in \Sigma^n$ and $t \in \Sigma^m$ if and only if there exist ℓ, ℓ' with $0 \leq \ell \leq |s| - 1$ and $0 \leq \ell' \leq |t| - 1$, such that x occurs in s at position ℓ and in t at position ℓ'.*

Definition 21 (Occurrence pair) *A common motif x that occurs in s at position ℓ and in t at position ℓ' has an* occurrence pair *$\mathcal{P}_x = (\ell, \ell')$. Given an occurrence pair $\mathcal{P}$, we denote by $\mathcal{P} + i$ the set $\{(\ell + i, \ell' + i) \mid \mathcal{P} = (\ell, \ell')\}$.*

The above definition associates to a motif just a pair of occurrences independently from the possibility that the same motif occurs also elsewhere in s or t. In this sense, the occurrence pair is a *weak* version of the complete occurrence list of a common motif, and for this reason we call the resulting basis (defined below) a *weak basis.*

Definition 22 (w-maximal common motifs) *A common motif x is w-*maximal *(weak maximal) if and only if it has an occurrence pair $\mathcal{P}_x$ such that, for each common motif y with an occurrence pair $\mathcal{P}_y$ and integer d with $\mathcal{P}_y = \mathcal{P}_x + d$, we have that y occurs in x.*

Observe that the notion of maximality of a common motif is related to one (or more) of its occurrence pairs that, unlike occurrence lists, are not unique.

Example Let $s =$ `ATGATC` and $t =$ `ATCATG`. The common motif `ATC` has only the occurrence pair $(3, 0)$ for which it is w-maximal. The common motif `AT` has four occurrence pairs, namely $(0, 0)$, $(0, 3)$, $(3, 3)$, and $(3, 0)$, for which it is not w-maximal. For example, for the occurrence pair $(0, 3)$, there is the common motif `ATG` with the same occurrence pair and it does not occur in `AT`.

Since the occurrence list is here limited to a pair for all w-maximal motifs, there is no need to appeal to more restrictive properties such as the property of not being tiled by other w-maximal common motifs. Therefore, we suggest as a first notion of basis for the motifs common of s and t, the set of all w-maximal common motifs with the corresponding occurrence pairs. Such a basis, called a *weak basis*, has an interesting and surprisingly simple characterization which leads to an efficient algorithm for computing it. This is formalized in the following definitions and results.

Definition 23 (*CMerge*) *Let us assume that* $t[i] = \circ$ *for all* $i < 0$ *and* $i > |t| - 1$. *For all possible shifts* k *such that* $-(n-1) \le k \le m-1$, *let* c_k *be the string such that* $c_k[i] = s[i] \oplus t[i+k]\ \forall\ 0 \le i \le |s| - 1$. *We denote by* $CMerge_k$ *(common merge) the string obtained from* c_k *by removing all don't cares to the left of the leftmost solid character and to the right of the rightmost solid character.*

Example Given a string t, let us consider again $s =$ `ATGATC` and $t =$ `ATCATG`. We have that the non empty *CMerge*s are $CMerge_{-3} =$ `ATC`, $CMerge_0 =$ `AT`$\circ$`AT`, and $CMerge_3 =$ `ATG`.

The *CMerge*s are analogous to the merges introduced in Section 4.1, except that they are no longer obtained by superposing a string s with all the shifts of itself, but rather by superposing the longest string t with all possible shifts of the other string s. Notice that any common motif c has one occurrence in $CMerge_{(j-i)}$ for each one of its occurrence pairs $\mathcal{P}_c = (i, j)$.

Theorem 6 *The weak basis coincides with the set of distinct non empty CMerges.*

Proof: We first show that all non empty *CMerge*s are w-maximal common motifs. Let $c = CMerge_k$ be non empty. It has an occurrence pair $\mathcal{P}_c = (\ell, \ell + k)$. Let us assume that there exists another common motif y with an occurrence pair $\mathcal{P}_y$ such that $\mathcal{P}_y = \mathcal{P}_c + d$ for some integer d. As observed above, it must be that y occurs in $CMerge_k$ and thus in c.

We now show that all w-maximal common motifs are *CMerge*s. Let x be a w-maximal common motif with the occurrence pair $\mathcal{P}_x = (\ell, \ell')$. From the remark above, we have that x occurs in $CMerge_{(\ell'-\ell)}$. Moreover, $CMerge_{(\ell'-\ell)}$ has the occurrence pair (ℓ, ℓ'), which is equal to $\mathcal{P}_x$ (and thus in particular it is a d-shift with $d = 0$), and by definition of w-maximality of x it must be that $CMerge_{(\ell'-\ell)}$ occurs in x, which implies that $x = CMerge_{(\ell'-\ell)}$. $\bowtie$

As a consequence, the size of the weak basis is linearly bounded (there are at most as many motifs in the weak basis as distinct *CMerge*s), and computing it is as easy as computing the set of distinct non empty *CMerge*s, which can obviously be done in $O(n(m+n))$ time.

6.2 Strong Basis

The reason for the efficiency of the algorithm that computes the weak basis lies on the fact that there is no need to compute all the occurrences of the *CMerges*, as this is not required in the output nor is needed for a selection like in the case of tiling motifs. Nevertheless, a complete list of all the occurrences of a common motif may be required by certain applications. For this reason, we now define a new notion of occurrence list for common motifs, and hence a corresponding new notion of maximality and basis.

Definition 24 (Common motifs occurrence list) *A common motif x has as* occurrence list *the pair of lists $\mathcal{CL}_x = (\mathcal{L}_{(x,s)}, \mathcal{L}_{(x,t)})$ where $\mathcal{L}_{(x,s)}$ (resp. $\mathcal{L}_{(x,t)}$) is the list of all the occurrences of x in s (resp. t). Given an occurrence list $\mathcal{CL} = (\mathcal{L}_s, \mathcal{L}_t)$, we denote by $\mathcal{CL} + i$ the pair of lists $(\mathcal{L}_s + i, \mathcal{L}_t + i)$.*

Observe that for a given common motif x, its occurrence list is unique, differently from the case of occurrence pairs. In fact, any common motif with occurrence list $\mathcal{CL}_x = (\mathcal{L}_{(x,s)}, \mathcal{L}_{(x,t)})$ has an occurrence pair (i, j) for each $i \in \mathcal{L}_{(x,s)}$ and $j \in \mathcal{L}_{(x,s)}$. As a direct consequence of this, we have that also for occurrence lists of common motifs, there is a relation with the *CMerges*. Namely, each common motif x with occurrence list $\mathcal{CL}_x = (\mathcal{L}_{(x,s)}, \mathcal{L}_{(x,t)})$ occurs in $CMerge_{(j-i)}$ for each $i \in \mathcal{L}_{(x,s)}$ and $j \in \mathcal{L}_{(x,s)}$. Based on this more complete version of occurrence set for common motifs, a new notion of maximality can be defined, as well as a notion of tiling motifs.

Definition 25 (Maximal common motifs) *A common motif x is* maximal *if and only if for each common motif y and integer d such that $\mathcal{CL}_y = \mathcal{CL}_x + d$, we have that y occurs in x (and thus $d \geq 0$).*

Definition 26 (Tiling common motifs) *Given two occurrence lists of common motifs $\mathcal{CL}_1 = (\mathcal{L}_1, \mathcal{L}'_1)$ and $\mathcal{CL}_2 = (\mathcal{L}_2, \mathcal{L}'_2)$, let $\mathcal{CL}_1 \cup \mathcal{CL}_2 = ((\mathcal{L}_1 \cup \mathcal{L}_2), (\mathcal{L}'_1 \cup \mathcal{L}'_2))$. A maximal common motif x is a* tiling common motif *if and only if, for any maximal common motifs y_1, y_2, ..., y_k and integers d_1, d_2, ..., d_k such that $\mathcal{L}_x = \cup_{i=1}^{k}(\mathcal{L}_{y_i} + d_i)$, motif x is one y_i. When, instead, all y_i are different from x, pattern x is said to be a* tiled common motif. *We also say that the common motifs y_i, y_2, ..., y_k* tile *x.*

Example Let us consider again the strings s = `ATGATCATG` and t = `ATCATGATC` and their maximal common motif `ATG` with occurrence list $\mathcal{CL}_{\texttt{ATG}} = (\{0, 6\}, \{3\})$. This motif is maximal but not tiling because it is tiled by the maximal common motifs `ATCATG` and `ATGATC`. In fact, the latter are both maximal because they are equal to $CMerge_{-3}$ and $CMerge_3$, respectively; moreover, since $\mathcal{CL}_{\texttt{ATCATG}} = (\{3\}, \{0\})$ and $\mathcal{CL}_{\texttt{ATGATC}} = (\{0\}, \{3\})$, we have that $\mathcal{CL}_{\texttt{ATG}} = (\mathcal{CL}_{\texttt{ATCATG}} + 3) \cup \mathcal{CL}_{\texttt{ATGATC}}$.

The second notion of basis for common motifs that we suggest is that set of all tiling common motifs, and we call it *strong basis*. Similarly to the case of repeated motifs, the following results hold[3]:

[3] We omit the proofs either because they are similar to those for repeated motifs or because they are trivial consequences of previous results.

- For all k such that $-(n-1) \leq k \leq m-1$, $CMerge_k$ is a maximal common motif.
- Each maximal common motif is also w-maximal for a suitable occurrence pair.
- Each tiling common motif is a $CMerge_k$ for some k with $-(n-1) \leq k \leq m-1$. Nevertheless, in general not all *CMerge*s are tiling common motifs.
- The strong basis for common motifs contains $O(n+m)$ motifs.
- The strong basis for common motifs is symmetric.

The strong basis can be computed by an algorithm similar to the one shown in [13] for repeated motifs. Indeed, we can prove that for each tiling common motif the list $\mathcal{CL}_x$ coincides with the list CT_x of the occurrence pairs that can be directly recovered from the *CMerge*s that are equal to x. Furthermore, the sum of all these occurrences is linearly bounded. This allows an efficient detection of non tiling common motifs among the *CMerge*s. In conclusion, we have that the strong basis can be computed in the following way. First, the set of all the distinct *CMerge*s can be computed in $O(n(n+m))$ time and there are $O(n+m)$ of them. Second, finding all their occurrences and discarding those such that $CT \neq \mathcal{CL}$ takes $O((n+m)m \log m) = O(m^2 \log m)$ time. Finally, discarding the remaining non tiling motifs can be performed in $O((m+n)^2)$ time using two bit vectors of size $n+m$. The overall time complexity is thus in $O(m^2 + (n+m)m \log m + (m+n)^2) = O(m^2 \log m)$.

Acknowledgments

We wish to thank Laxmi Parida for sending us a copy of reference [1].

References

[1] A. Apostolico and L. Parida. Incremental paradigms of motif discovery. Unpublished, 2002.

[2] A. Apostolico and L. Parida. Compression and the wheel of fortune. In *IEEE Data Compression Conference (DCC'2003)*, pages 143–152, 2003.

[3] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval.* Addison Wesley, May 1999.

[4] A. Brazma, I. Jonassen, I. Eidhammer, and D.R. Gilbert. Approaches to the automatic discovery of patterns in biosequences. *Journal of Computational Biology*, 5(2):279–305, 1998.

[5] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification.* Wiley Interscience, 2000.

[6] T. Imielinski and H. Mannila. A database perspective on knowledge discovery. *Communications of the ACM*, 39:58–64, 1999.

[7] M. Lothaire. *Combinatorics on Words.* Cambridge University Press, 1997.

[8] M. Lothaire. *Algebraic Combinatorics on Words*. Cambridge University Press, 2002.

[9] L. Parida, I. Rigoutsos, A. Floratos, D. Platt, and Y. Gao. Pattern Discovery on Character Sets and Real-valued Data: Linear Bound on Irredundant Motifs and Efficient Polynomial Time Algorithm. In *Proceedings of the eleventh ACM-SIAM Symposium on Discrete Algorithms (SODA 2000)*, pages 297–308, 2000.

[10] L. Parida, I. Rigoutsos, and D. Platt. An output-sensitive flexible pattern discovery algorithm. In A. Amir and G.M. Landau, editors, *Combinatorial Pattern Matching*, volume 2089 of *LNCS*, pages 131–142. Springer-Verlag, 2001.

[11] J. Pelfrêne, S. Abdeddaïm, and J. Alexandre. Un algorithme d'indexation de motifs approchés. In *Journée Ouvertes Biologie Informatique Mathématiques (JOBIM)*, pages 263–264, 2002.

[12] J. Pelfrêne, S. Abdeddaïm, and J. Alexandre. Extracting approximare patterns. In *Combinatorial Pattern Matching*, volume 2676 of *LNCS*, pages 328–347. Springer-Verlag, 2003.

[13] N. Pisanti, M. Crochemore, R. Grossi, and M.-F. Sagot. A basis for repeated motifs in pattern discovery and text mining. Technical Report IGM 2002-10, Institut Gaspard-Monge, University of Marne-la-Vallée, July 2002.

[14] N. Pisanti, M. Crochemore, R. Grossi, and M.-F. Sagot. Bases of motifs for generating repeated patterns with don't cares. Technical Report TR-03-02, Dipartimento di Informatica, University of Pisa, January 2003. Submitted to journal.

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

[16] M.-F. Sagot and A. Viari. A double combinatorial approach to discovering patterns in biological sequences. In D. Hirschberg and G. Myers, editors, *Combinatorial Pattern Matching*, volume 1075 of *LNCS*, pages 186–208. Springer-Verlag, 1996.

[17] M.-F. Sagot, A. Viari, and H. Soldano. Multiple comparison: a peptide matching approach. *Theoret. Comput. Sci.*, 180:115–137, 1997.