Università degli studi di Padova

Dipartimento di Ingegneria dell' Informazione

Corso di laurea Magistrale in Ingegneria informatica

Mining motifs in Temporal networks

Laureando
Ilie Sarpe

Relatore
Prof. Fabio Vandin

Anno Accademico 2018/2019

Abstract

Temporal networks are mathematical tools used to model complex systems which embed the temporal dimension; the ability of such networks to represent time, makes them useful in a huge variety of fields ranging from biology to physics. Counting little *subnetworks* of interest, called *motifs*, is one of the key tasks in the analysis of the temporal networks, since the counts of the motifs can characterize in a unique way a temporal network and it's functions. The increasing production and the availability of big temporal network datasets require efficient, scalable and rigorous techniques for extracting useful information from such large datasets.

In this thesis we address the problem of counting motifs in temporal networks and we provide several algorithms for such problem. We provide a new exact parallel algorithm, obtained from the combination of two existing techniques, which is both scalable and efficient in practice. Such algorithm provides the exact number of temporal motifs in a temporal network.

Exactly counting the number of motifs in a large network may be computationally infeasible, thus we address the problem of approximating such count with rigorous guarantees. For this purpose, we present the definition of (ϵ, η)-approximation for the problem of counting temporal motifs, and we provide, to the best of our knowledge, the first rigorous sampling algorithms ever devised for such task. We rigorously prove their guarantees, their variance, the running time and we provide different bounds on the number of samples s required to achieve the desired approximation factor.

We then tested all of these techniques on real world datasets, comparing them to the state of the art techniques in such field, evaluating their efficiency, scalability and accuracy.

Ringraziamenti

Vorrei ringraziare prima di tutto mia mamma Maria, per avermi aiutato a portare avanti la carriera accademica e per aver avuto fiducia nelle mie scelte.

Un grazie particolare va a Gaetano, che è sempre presente per supportarmi o un fornirmi un consiglio quando ne ho bisogno ma anche solo per vedere una serie tv.

Ringrazio Andrea, compagno di università con il quale abbiamo affrontato questi anni tra progetti e corsi più disparati, per avermi accompagnato in questo percorso.

Ringrazio Michele, per avermi spinto ad assistere a più concerti live accompagnandomi in questi e per avermi fatto appasionare al mondo della cucina.

Infine vorrei ringraziare il Prof. Fabio Vandin, senza il quale questo lavoro non sarebbe stato possibile, voglio ringraziarlo per avere creduto nelle mie capacità e per la passione e la motivazione che mi trasmette.

Contents

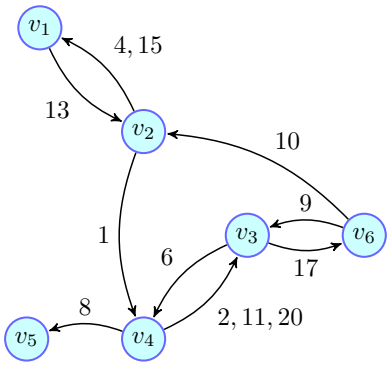
Chapter 1

Introduction

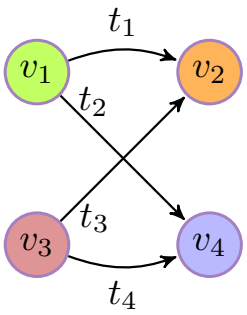
Large amount of data are produced everyday from different systems, such as social and biological networks, internet of things, sensor networks, peer to peer networks and many other systems. A key challenge in the *data mining* field is to extract useful information from such data, to characterize and understand better the behaviour and the rules of such complex systems [2]. Many approaches have been proposed for the extraction of useful information from large amount of data, which are fundamental tasks in data mining, such as pattern mining, clustering, similarity search, and graph mining [8]. Usually, when data come from networks the system is modelled as a graph, which is a mathematical abstraction for studying complex systems that helps to characterize the system's behaviour and to compute meaningful quantities that give us useful informations.

In data mining, after modelling the system as a graph, one of the fundamental primitives is to identify small graphs, called *graphlets* or *motifs*, which impose a certain topology and are fundamental for the comprehension of the network; e.g., counting triangles is fundamental for computing the clustering coefficient [14], moreover the number of motifs may be used to compare different networks, for example in biology [13]. Identifying motifs is thus a fundamental primitive and many techniques have been proposed in literature to address such topic [6]. While there exist exact techniques which do not scale on large datasets and approximated techniques which can or cannot provide rigorous guarantees on the quality of the approximation [3, 14], unfortunately all of these techniques have been devised for *static networks*, which do not embed the temporal dimension.

Temporal networks or *temporal graphs* are a mathematical abstraction for representing complex systems, and embedding in such representation the temporal dimension. Many definitions have been proposed in literature [1, 4, 5], in this thesis we will refer to a temporal graph following the definition of [7, 9, 12]. Informally, we may say that a temporal graph is a sequence of edges where each edge has an additional information, that is the timestamp of



(a) Temporal graph.



(b) Temporal motif.

Figure 1.1: (a) Example of temporal graph, each edge reports a timestamp corresponding to the time of the event which the edge represents. (b) Example of temporal motif, a small subgraph which imposes a topology of interest and a given order to the sequence of edges, in particular this is a 4-node 4-edge temporal motif, known as "bi-fan motif" [9].

the event which the edge represents. For example, in an e-mail network each edge is represented through the users which send and receive the mail and the timestamp of the mail; one example of a temporal network is reported in figure (1.1a).

Accounting for the temporal dimension is natural, since all the systems of interest already mentioned present a temporal dimension which may give very useful information on their structure, but computationally this makes thing much more difficult. In particular, it has been proved that the techniques developed for static graphs are not easily adaptable for temporal networks [4, 10, 12]. Furthermore, the large amount of data produced every day requires efficient, scalable and rigorous techniques for handling very large datasets.

In order to understand and discuss the temporal networks and the state of the art techniques in such field, we need to formalize better the intuition already given. The next section reports the basic definitions needed in this thesis.

1.1 Basic Definitions

In this section we present the basic definitions needed in the presentation of this thesis, and we define rigorously the problem of mining motifs in temporal networks.

Definition 1. A *temporal graph* is a pair $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ where, $\mathcal{V} = \{v_1, \ldots, v_n\}$ and $\mathcal{E} = \{(u, v, t) : u, v \in \mathcal{V}, u \neq v, t \in \mathbb{R}^+\}$ with $|\mathcal{V}| = n$ and $|\mathcal{E}| = m$.

We may also denote $\mathcal{V} = \{v_1, \ldots, v_n\}$ with the set $\mathcal{V} = \{1, \ldots, n\}$ of the first

n natural numbers. We also assume the edges in $\mathcal{E}$ to be sorted by increasing timestamps and the timestamps to be unique, this is not a loss of generality since the methods we are going to discuss can also handle the cases where there can be edges with the same timestamps.

Each directed edge $e = (u, v, t) \in \mathcal{E}$ carries the temporal information, represented as a timestamp t in $\mathbb{R}^+$, for the interaction between the nodes $u, v \in \mathcal{V}$ at time t, as reported in figure (1.1a).

We now define the concept of *temporal motif* following the definition introduced by Liu et al. in [9]:

Definition 2. A k-node l-edge *temporal motif* is a pair $M = (\mathcal{K}, \sigma)$ where $\mathcal{K} = (\mathcal{V}_\mathcal{K}, \mathcal{E}_\mathcal{K})$ is a static and connected (multi)graph where $\mathcal{V}_\mathcal{K} = \{v_1, \ldots, v_k\}$, $\mathcal{E}_\mathcal{K} = \{(u, v) : u, v \in \mathcal{V}_\mathcal{K}, u \neq v\}$ s.t. $|\mathcal{V}_\mathcal{K}| = k$ and $|\mathcal{E}_\mathcal{K}| = l$ and σ is an ordering of $\mathcal{E}_\mathcal{K}$.

The temporal motif $M = (\mathcal{K}, \sigma)$ can be denoted with $(u_1, v_1), \ldots, (u_l, v_l)$ i.e., the edges of $\mathcal{E}_\mathcal{K}$ between nodes of $\mathcal{V}_\mathcal{K}$ ordered according to σ. An example of temporal motif is reported in figure (1.1b). Informally we can say that, the temporal motif M represents the schema for which we want to count all the occurrences in the graph $\mathcal{T}$ within a given timespan $\delta \in \mathbb{R}^+$. In order to formalize this intuition we present the following definition:

Definition 3. Given a temporal graph $\mathcal{T} = (\mathcal{V}, \mathcal{E})$, a temporal motif $M = (\mathcal{K}, \sigma)$, and $\delta \in \mathbb{R}^+$, a time ordered sequence $S = (u_1', v_1', t_1'), \ldots, (u_l', v_l', t_l')$ of l unique temporal edges from $\mathcal{T}$ is a *δ-instance* of the temporal motif $M = (u_1, v_1), \ldots, (u_l, v_l)$ if:

1. there exists a bijection f on the vertices such that $f(u_i') = u_i$ and $f(v_i') = v_i$, $i = i, \ldots, l$ and
2. the edges all occur within δ time, i.e., $t_l' - t_1' \leq \delta$.

Note that here we slightly abuse the notation saying that the edges come from $\mathcal{T}$ instead of $\mathcal{E}$. A δ-instance is thus, informally, a sequence of edges from the original graph which has the same topology of the motif M and did not violate an additional constraint on the temporal dimension, i.e., point 2 in Definition 3.

We define the set of all the δ-instances as follows.

Definition 4. The *set of all δ-instances* of the motif M in $\mathcal{T}$ is $\mathcal{U} = \{U : U$ is a δ-instance of M from a sequence of edges from $\mathcal{T}\}$, we denote the cardinality of $\mathcal{U}$ with $|\mathcal{U}| = C_M$.

Given one δ-instance of the motif M, the following definition will be useful in this thesis.

Definition 5. For each δ-instance of M namely, for each $U = (u_1^U, v_1^U, t_1^U), \ldots, (u_l^U, v_l^U, t_l^U) \in \mathcal{U}$ the *motif duration* is defined as $\Delta(U) \triangleq t_l^U - t_1^U$. We also denote t_1^U and t_l^U respectively as the *starting time* and *ending time* of the instance $U \in \mathcal{U}$.

Given these definitions we now can formalize the goal of temporal motif mining.

Goal. Given a temporal graph $\mathcal{T}$, a temporal motif $M = (\mathcal{K}, \sigma)$, and $\delta \in \mathbb{R}^+$, we want to compute C_M i.e., the exact number of δ-instances of motif M in the temporal graph $\mathcal{T}$.

Other useful definitions are the following ones.

Definition 6. Given a temporal graph $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ we say that $G_d = (V_d, E_d)$ is the *directed static subgraph associated with* $\mathcal{T}$ or simply the *directed static subgraph of* $\mathcal{T}$ if $V_d = \mathcal{V}$ and $E_d = \{(u, v) | \exists t : (u, v, t) \in \mathcal{E}\}$.

Definition 7. Given a temporal graph $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ and given $G_d = (V_d, E_d)$ the directed static subgraph of $\mathcal{T}$ we say $G_u = (V_u, E_u)$ to be the *undirected static subgraph associated with* $\mathcal{T}$ or simply the *undirected static subgraph of* $\mathcal{T}$ if $V_u = V_d = \mathcal{V}$ and $E_d = \{\{u, v\} | (\exists (u, v) \in E_d) \vee (\exists (v, u) \in E_d)\}$, where with $\{a, b\}$ we denote an undirected edge between a and b.

We may now look at a brief summary which describes the previous works in the field of static and temporal motif mining.

1.2 Related Work

In this section we review some of the main works in the field motif mining, both in static and temporal networks.

As mentioned in the introduction, counting motifs in graphs is a basic primitive in data mining, thus many such techniques exist for static graphs. Since exact approaches are usually not practicable, many approximated approaches exist [3, 14], which can also provide rigorous guarantees on the quality of the approximation. The number of motifs in a network may be used to characterize the network's behaviour, i.e., through the computation of the clustering/local closure coefficient or the distribution of the motifs in a network [13, 18, 19].

For the temporal networks instead not so many techniques for the problem of counting motifs exist, the main works which follow the direction of this thesis are [9, 12] which we will discuss in Chapter 2. A paper which goes in a similar direction but employs a different definition of a *motif* is the paper by Kovanen et al. [7]. Their definition requires the edges of the motif instance to be consecutive, i.e., no other temporal edge may occur in between the events a motif; such definition is less general than the one used in this thesis and

may simplify some tasks, such as triangle counting which may be done in linear time under such definition [9]. Finally, some exact routines have been proposed such as [10, 15]. In the paper of Mackey et al. [10], the authors devised an exact routine for enumerating all the motifs instances of a given motif M. In the work of Sun et al. [15] they devised a new algorithm for counting the motifs instances to address a slightly different problem, that is to compute the most frequent motifs in a temporal graph. As for the static networks, several applications there exist that use the counts of the temporal motifs such as [16], where the authors use a metric based on the number of motifs to classify different temporal networks. Another interesting application comes from [17], where the authors used a slightly different definition of temporal graph, adding to each edge an interval instead of a timestamp, to represent long interactions; then a method to find interesting cliques in the IP traffic is exploited.

Since the topic of temporal networks is new, not so much techniques for mining motifs exist as we presented, moreover very few techniques adopt the definitions we are using in this thesis, for an extensive review of other definitions of temporal network and many other tasks related to such topics we refer the reader to [1, 4, 5].

1.3 Our contributions

Our main contributions to the field of motif mining in temporal networks is the development of the following algorithms:

- A first rigorous and scalable sampling algorithm for approximating the motif counts in large temporal networks;
- An improved rigorous and scalable sampling algorithm for approximating the motif counts in large temporal networks;
- A parallelizable scalable technique for counting the number of motifs in a temporal network exactly.

The approximation algorithms are presented in Chapter 3, where we analyse such techniques, proving the correctness and the approximation factor. For the first algorithm we use an analysis based on the Hoeffding inequality, while for the improved algorithm in addition to the analysis with the Hoeffding Bound we developed a more involved analysis based on the tool of Martingales. Moreover we analyse the variance of the estimate used in this improved algorithm obtaining two different bounds. We also analyse the asymptotic complexity of both the algorithms. To the best of our knowledge, these are the first rigorous sampling algorithms for approximating the counts of motif in a temporal network ever devised.

The exact scalable routine is reported in Chapter 4, as we will show such approach is based on the combination of the works of [15] and [10]; which together may lead to a very scalable and efficient exact algorithm for enumerating and thus counting temporal motifs.

All such techniques were then implemented in C++ and tested on several datasets from the SNAP[1] collection. The results of the tests and the discussion of all the methods is then reported in Chapter 5.

1.4 Outline of the thesis

The thesis is organized as follows, in Chapter 2 we present the state of the art techniques for mining temporal motifs, that are more similar to our work. Chapter 3, describes the new sampling algorithms we developed for approximating the count of a motif M in a temporal network $\mathcal{T}$. Chapter 4 presents the parallel exact algorithm we developed for counting temporal motifs. Chapter 5 reports all the tests we performed comparing both the state of the art techniques for counting temporal motifs and our algorithms. The last chapter, Chapter 6 reports all the conclusions of our work and possible future directions.

[1] `https://snap.stanford.edu/data/#temporal`

Chapter 2

Previous Approaches

As we mentioned in the introduction, in the related work section, the main works that follow the direction of the algorithms we devised in Chapter 3 and Chapter 4, are the works of Paranjape et al. [12] and Liu et al. [9]. In this chapter we review such works, that will better provide an insight of the state of the art techniques for mining motifs in temporal networks. In the first section we present the work of Paranjape et al., discussing the definitions and the techniques they introduced, then we discuss the work of Liu et al., which is more similar to the work we will present in Chapter 3.

2.1 Work of Paranjape et al.

The work of Paranjape et al. was the first work to adopt the definition of temporal motif as we already presented in definition 2, they also provided different exact algorithms for counting temporal motifs based on the *dynamic programming* technique. In particular they provided:

- An algorithmic framework and an exact routine for counting all the δ-instances of different k-node l-edge temporal motif to be used in such framework;
- An algorithmic framework that can be adapted to count all the δ-instances of the 3-node 3-edge *star motifs* or all the δ-instances of the 3-node 3-edge *triangle motifs*.

For the sake of clarity, we can see in figure 2.1 the motifs with a grey background which are the *triangle motifs*, the motifs with green background that are the 2-*node motifs* and all the other motifs that are the *star motifs*.

2.1.1 General Schema

Now we present the first general schema designed by Paranjape et al., in particular an idea of how the algorithm they devised are designed, all the

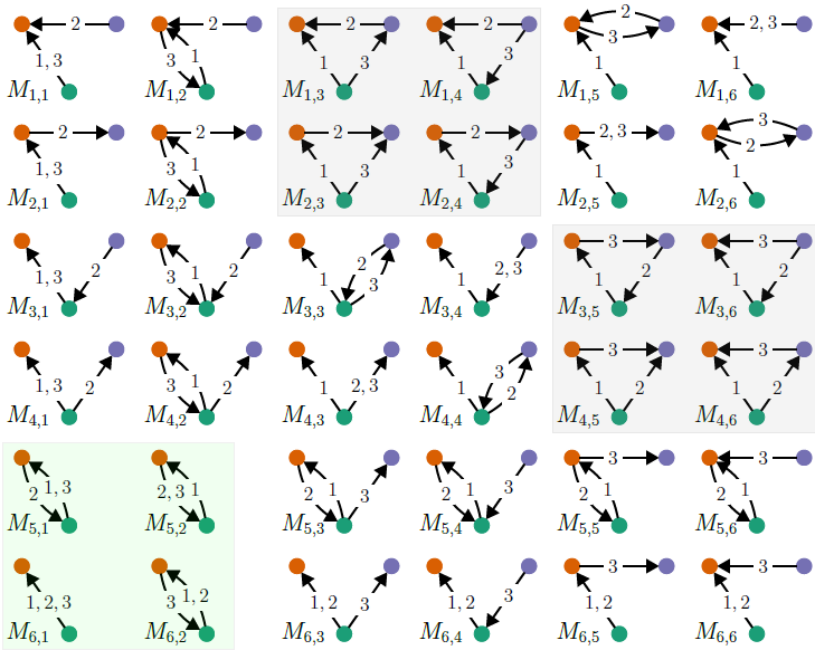


Figure 2.1: All the possible motifs up to 3-nodes and 3-edges, grid from [12].

Algorithm 1: General counting schema adopted by Paranjape et al.

Input: $\mathcal{T} = (\mathcal{V}, \mathcal{E}), \delta \in \mathbb{R}^+, M = (\mathcal{K}, \sigma)$.
Output: C_M exact number of δ-instances of M in $\mathcal{T}$.

1 $H \leftarrow$ staticDirectedGraph(M)
2 $G_d \leftarrow$ staticDirectedGraph($\mathcal{T}$)
3 $H_1, \dots H_u \leftarrow$ static instaces of H in G_d
4 $C_M \leftarrow 0$
5 **for** $i = 1, \dots, u$ **do**
6 $\quad S \leftarrow$ all the edges of $\mathcal{E}$ between pair of nodes forming an edge in H_i
7 $\quad S' \leftarrow$ sort(S)
8 $\quad C_M \leftarrow C_M +$ exactCount(S', δ, M)
9 **return** C_M

details may be found in the original paper [12]. The general procedure that they adopt is the one described in Algorithm 1. In line 1 they get H, the directed static graph of motif M and in line 2 they get the static directed graph of $\mathcal{T}$. In line 3 they compute all the static instances of H in G_d. Up to this point they did not considered the temporal dimension. This is done in the inner cycle (lines 5-8) where for each static instance, they gather the respective temporal edges from $\mathcal{T}$, i.e., those temporal edges that "generated"

the static dynamic edges of $H_i, i = 1, \ldots, u$. Such sequence is used as a candidate for extracting a part of the true count C_M, avoiding to use the routine "exactCount" on the whole edges of $\mathcal{E}$. Along with this Algorithm 1, the authors provided three different algorithms that can be used as exact routines which use the powerful tool of *dynamic programming*; we now discuss such exact routines.

2.1.2 Counting all the δ-instances with l edges

The first procedure Paranjape et al. devised is based on the following simple idea, that each motif it may be represented as a sequence of edges. The authors use such idea to define the data structure "counts" which assigns for a sequence of static directed edges a count, in particular:

- counts$[e_1, \ldots, e_r]$ gives the counts of the sequence of edges $e_1, \ldots, e_r$ in the current portion of edges being examined if $r < l$;
- counts$[e_1, \ldots, e_l]$ keeps the counts of the sequence of edges $e_1, \ldots, e_l$ during the whole procedure.

How this data structure is used is presented in Algorithm 2, the idea is to span the input sequence S looking at the subsequence of temporal edges $(e^d_{start}, t_{start}), \ldots, (e^d_{end}, t_{end})$ such that $t_{end} - t_{start} \leq \delta$, in such sequences we update the data structure counts. It is quite clear that, at the end of the algorithm, we can access from the data structure counts, the count of the sequence of edges that represent the motif M and return such result. Thus lines 1-3 initialize the data structure and the indexes, from line 4 the input sequence is spanned, in lines 5-7 if $t_{end} - t_{start} > \delta$ the value of start is increased and counts is updated properly.

Algorithm 2: Extract of algorithm 1 by [12]

Input: $S = (e^d_1, t_1), \ldots (e^d_L, t_L)$ sorted list of edges, $\delta \in \mathbb{R}^+, M$ temporal motif.
Output: C^S_M exact number of δ-instances of M in S.

```
1  start ← 1
2  end ← 1
3  counts ← empty structure
4  while end ≤ L do
5      while t_end − t_start > δ do
6          DecrementCounts(e_start)
7          start ← start+1
8      IncrementCounts(e_end)
9      end ← end+1
10 C^S_M ← counts[M]
11 return C^S_M
```

Instead if $t_{end} - t_{start} \leq \delta$ then end is increased and counts is updated concordantly. After all the edges have been spanned, then the number of δ-instances of motif M in the input sequence S may be returned from the algorithm. The extra routines may be found in the original work, we do not present such routines since they are not fundamental for understanding the algorithm. Observe that such algorithm may be used to count all the occurrences of an l-edge motif, and not only those of a specific motif which makes it a powerful tool. Note that such algorithm counts the δ-instances in the input sequence and do not enumerate them, which is a harder problem.

The complexity of such algorithm depends on the number of sequences accounted for, in the data structure counts which may be up to $O(|H|^l)$ in the general case where $|H|$ is the number of edges in the static directed graph of M. The overall complexity of such routine is up to $O(|H|^l|S|)$, if one restricts the sequences in counts to be contiguous, then the complexity becomes $O(l^2|S|)$ since only $O(l^2)$ may be active for contiguous sequences. The complexity of the algorithm 1 is thus $O(\text{staticEnumeration} + |H|^l \sum_S |S|)$ using algorithm 2 as subroutine and counting all the possible l-edges motifs in such subroutine. If instead such algorithm is used to count only the δ-instances of the motif of interest, then the routine has complexity $O(\text{staticEnumeration} + l^2 \sum_S |S|)$. It is interesting to note that the routine of Algorithm 1 may be parallelized, launching all the iterations of cycle in lines 5-8 in parallel the running time becomes $O(\text{staticEnumeration} + |H|^l \max_S\{|S|\})$ in the general case. Observe that if the algorithm is used to count motifs with 2 nodes, then $|H| \leq 2$ since at most two edges may connect two nodes in the *static directed* graph of M, moreover $\sum_S |S| = O(m)$ since for each pair of nodes the Algorithm 1 gathers the temporal edges which connects them; thus also no enumeration is required. Thus in such case the complexity is limited by $O(2^l m)$ and if l is small, typically up to 3, then the algorithm is linear up to constant factors.

Such algorithm is thus used to count 2-nodes motifs up to 3-edges, but it achieves poor performances on other motifs, as also the authors discuss in their work.

2.1.3 Improved exact routines

The authors provided much more complicated routines for counting 3-nodes and 3-edges star motifs which use different data structures and a revised procedure; this leads to an algorithm with a complexity $O(m)$, which is linear in the number of edges of the whole temporal graph. Along with such routine, a fast algorithm for counting triangle motifs is presented which achieves a complexity of $O(\text{staticEnumeration} + m\sqrt{\tau})$, where staticEnumeration is the complexity of enumerating all the triangles in G_u (recall definition 7) the undirected static graph of $\mathcal{T}$ and τ is the number of static triangles in G_u. Such complexity is significantly better than the $O(\text{staticEnumeration} + m\tau)$ that results from using the Algorithm 1 with Algorithm 2 to count the triangle

motifs.

We conclude observing that the algorithms developed by Paranjape et al. have efficient asymptotic running time for motifs with at most 3-nodes and 3-edges, but they cannot scale for general k-node l-edge temporal motif. Furthermore, as also the authors specify, their algorithms cannot be adapted to enumerate the temporal motifs in a temporal network.

2.2 Work of Liu et al.

In their work [9], Liu et al. adopted the same definitions of temporal graph and temporal motif as Paranjape et al., and they discussed the following topics,

- A proof of *NP*-hardness of counting a *star temporal motif*;
- The first sampling-based technique for approximating a count C_M of a motif M in a temporal network $\mathcal{T}$ with a count C'_M hopefully not so distant from C_M;
- Adapted such sampling technique to different algorithms;
- Developed an exact routine for counting a specific 2-node 3-edge motif, such motif reported in figure (2.2b).

Now we discuss each of these contributions.

2.2.1 Proof of *NP*-hardness

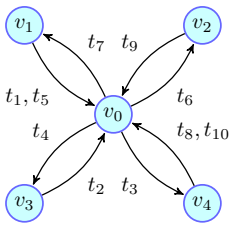
We begin with a definition,

Definition 8. A $\bar{k}$*-temporal star* is a temporal motif where the multigraph is connected and has $k = \bar{k} + 1$ nodes, namely $\{v_0, \ldots, v_{\bar{k}}\}$, with edges $(u_i, v_i), i = 1, \ldots, l$ where either u_i or v_i is v_0, $i = 1, \ldots, l$.

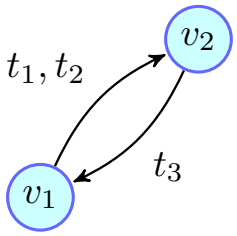
An example of such temporal motif is reported in figure (2.2a). Observe that counting such motifs in static graphs may be done in polynomial time since given one node u, it's degree d_u, and $k \in \mathbb{N}$, u is the center of $\binom{d_u}{k}$ $(k+1)$-node stars. Liu et al. proved that the same problem on temporal graphs is *NP*-hard. They defined the problem as follows,

Problem. Given a temporal graph $\mathcal{T}$, a $\bar{k}$-temporal star S, and a time span δ, the K-STAR-MOTIF problem asks if there exists at least one δ-instance of S in $\mathcal{T}$.

Proving that K-STAR-MOTIF is *NP*-hard is done reducing K-CLIQUE to such problem and the proof may be found in [9], the interesting thing is that such theorem shows how mining motifs in temporal networks may be much more difficult than mining motifs in static networks even if the *topologies* are the same.



(a) Star Motif.



(b) A 2-mode 3-edge motif.

Figure 2.2: (a) $\bar{k}$-star motif with $\bar{k} = 4$. (b) temporal motif for which [9] developed an exact algorithm.

2.2.2 Sampling Framework

Motivated from the fact that counting exactly temporal motifs requires a lot of computational resources, and from the availability of large temporal networks, Liu et al. introduced the first sampling framework for approximating the count C_M of all the δ-instances of a temporal motif M in a temporal network $\mathcal{T}$ with a number C'_M. Their sampling procedure employs the power of randomization. The basic idea is to look at the temporal dimension of the edges in $\mathcal{E}$, in such dimension all the edges are distributed from $t_1, \dots, t_m$ the time stamps of the different edges; with such image in mind the idea of Liu et al. is to partition the temporal dimension in different non-overlapping intervals of length $c\delta, c > 1$ which they define as follows, given s a *random shift* in $\{-c\delta + 1, \dots, 0\}$ at random,

$$\mathcal{I}_s = \{[s + (j-1)c\delta + j \cdot c\delta - 1], j = 1, 2, \dots\}.$$

Then clearly each interval of the set $\mathcal{I}_S$ contains a portion of edges of $\mathcal{E}$, moreover since the intervals are of length $c\delta$ many motifs will be included in such intervals. In their paper the authors show that counting such motifs and weighting each instance with the inverse of the probability of being accounted in some interval, leads to an unbiased estimate of C_M. In order to avoid to sample all the intervals the authors proposed to use the importance sampling, which allows to choose each sample with some probability, while keeping the unbiasedness of the estimate. The variance of the estimate before the application of the importance sampling is bounded by $\frac{1}{c-1}C_M^2$, while the estimate that uses the importance sampling has an increased variance due to such technique.

The algorithmic framework is presented in Algorithm 3, after the procedure selects a random shift in line 2, it spans all the set $\mathcal{I}_s$ and with a probability $q_j = r|I_j|/|\mathcal{E}|$ they choose if to look at the j-th interval (lines 3-4), where r is an hyper-parameter used to fix the situation where $|I_j|/|\mathcal{E}| << 1$, but which also plays a role in the weight of each motif counted. If the interval is

Algorithm 3: Algorithm 1 by [9]

Input: $\mathcal{T} = (\mathcal{V}, \mathcal{E}), M, \delta \in \mathbb{R}^+$, probabilities q, repetitions b, $c > 1$.
Output: C'_M estimate of C_M.

```
1  for a = 1, ..., b do
2      s ← randomInteger([−cδ + 1, ..., 0])
3      foreach I_j ∈ I_s (in parallel) do
4          if Uniform(0,1) ≤ q_j then
5              T_j ← {(u, v, t) ∈ E | t ∈ I_j}
6              {(count_i', Δ_i')} ← A(T_j, M, δ)
7              foreach (count_i', Δ_i') do
8                  Z_a ← Z_a + weighted(q_j, Δ_i', c, δ, count_i')
9  C'_M ← (1/b) Σ_{a=1}^{b} Z_a
10 return C'_M
```

to be accounted, then all the edges of $\mathcal{E}$ that fall in such interval are gathered (line 5) and all the δ instances in such set are mined using the algorithm $\mathcal{A}$. Then in the estimate, each motif is weighted with the inverse probability of being sampled (line 8). All the procedure from line 2 to 8 is repeated b times to reduce the final variance of a factor $\frac{1}{b}$, then the final estimate is computed as the arithmetic mean of all the partial b estimates (lines 9-10). The requirements on the algorithm $\mathcal{A}$ is not necessary to enumerate all the motifs but more specifically such routine has to produce the set $\{(count_i, \Delta_i)\}$ where $count_i$ is the number of instances of the motif with duration Δ_i in the specific interval.

Changing the algorithm $\mathcal{A}$ may have great impacts on the performances of such framework, the authors proposed the following combined algorithms:

- BT+S: The algorithm $\mathcal{A}$ is in this case is implemented with the backtracking algorithm of [10] which can enumerate all the possible δ-instances of a temporal motif M without any constraint on the number of nodes or edges of the motif;
- BT+PS: Has the same implementation of BT+S but now the cycle from line 3 is executed in parallel;
- EX23+S: The algorithm $\mathcal{A}$ is EX23, which is an algorithm designed by Liu et al. to suite the framework in Algorithm 3, in particular such combined algorithm may be used only to approximate the count of the motif in figure (2.2b).
- EX23+PS: a parallel version of the previous algorithm.

Huge impact on the performances has the choice of the algorithm, in particular we observe that the algorithm BT of [10] has an asymptotic

complexity of $O(|T_j|^l)$ where $|T_j|$ are the number of edges in the j-th sample and l are the number of edges of the motif. Then the BT+S has a complexity of $O(\sum_j |T_j|^l + |\mathcal{I}_s| C_M^{c\delta})$ where $C_M^{c\delta}$ is the maximum number of motifs in a temporal interval of length $c\delta$ and the term $|\mathcal{I}_s| C_M^{c\delta}$ arises in the global complexity from lines 7-8. Observe that in practice the complexity of BT+PS may be much lower than mining the whole graph which has complexity $O(m^l)$. While the BT+PS has a complexity of $O(\max_j\{|T_j|^l\} + C_M^{c\delta})$ if enough threads are available.

The complexity of EX23 is $O(\sum_{u,v} k_{u,v}^2)$ where $k_{u,v}$ is the number of temporal edges between nodes $u, v \in \mathcal{V}, u \neq v$, thus leading to a total complexity of $O(\sum_j \sum_{(u,v)\in T_j} k_{u,v}^2 + |\mathcal{I}_s| C_M^{c\delta})$ for the EX23+S algorithm and $O(\max_j\{\sum_{(u,v)\in T_j} k_{u,v}^2\} + C_M^{c\delta})$ for the EX23+PS. As an aside note, the authors claimed that the complexity of EX23 may be reduced, using some special tree data structures, to $O(\sum_{u,v} k_{u,v} \log(k_{u,v}))$ but, to the best of our knowledge, no description or implementation of such technique is currently available. We highlight the fact that EX23 and it's sampling version can count only one specific motif instance, motif in figure (2.2b); which makes it not very useful in practice with respect to BT and it's version, which can count all the instances of an arbitrary motif provided by the user.

Chapter 3

Novel Sampling Algorithms

Sampling approaches are required since, in large networks, computing exactly C_M is expensive in terms of both memory and time, thus approximating such count while providing rigorous guarantees on the quality of the approximation, is a key task for the motif counting problem. In this chapter we are going to introduce our sampling-based algorithms for the motif counting problem in particular the goal that we want to address is the following.

Goal of the approximation problem. Given a temporal graph $\mathcal{T}$, a temporal motif $M = (\mathcal{K}, \sigma)$, $\delta \in \mathbb{R}^+$,$(\epsilon, \eta) \in (0,1)^2$ we want to compute C'_M such that $\mathbb{P}(|C'_M - C_M| \geq \epsilon C_M) \leq \eta$, that is we want to obtain an ϵ-relative approximation to C_M with probability at least $1 - \eta$; where we recall that C_M is the exact number of δ-instances of M in $\mathcal{T}$. We call an algorithm that provides such guarantees an (ϵ, η)-*approximation algorithm.*

To the best of our knowledge the algorithms we are going to present are the first rigorous sampling techniques existing, for the problem of counting motifs in temporal networks. In this chapter we present:

- A first (ϵ, η)-approximation algorithm, for which we analyse the correctness and we provide a bound on it's sample size;
- An improved (ϵ, η)-approximation algorithm, for which we analyse the correctness and we provide a bound on it's sample size; we also provide two other bounds on the sample size using the tool of Martingales. We also analyse the variance of the estimate used in such approach.
- An analysis of the asymptotic running times of the methods mentioned above.

3.1 First Algorithm

The very first approach we introduce is based on sampling the temporal dimension as Liu et al. [9], but now gathering the interval of length $c\delta$ starting from a timestamp of an edge selected at random. This leads to the following intuitive steps:

1. Randomly choose with uniform probability a timestamp t_r of an edge $e \in \mathcal{E}$ such that t_r is between t_1 and t_{last} (both included), where $t_{last} = \arg\min_{t:(u,v,t)\in\mathcal{E}\wedge(t\geq t_m - c\delta)} \{|t - t_m + c\delta|\}$ for some $c > 1$;
2. gather all the edges $\{(u, v, t) \in \mathcal{E} : t_r \leq t \leq t_r + c\delta, c > 1\}$ from the original graph $\mathcal{T}$ and call the resulting sampled graph $\mathcal{T}_i$;
3. use an exact algorithm to count all the δ-instances of motif M in $\mathcal{T}_i$;
4. count the instances of motif M weighting each occurrence opportunely;
5. repeat the procedure for $i = 1, \ldots, s$ times in order to achieve the desired accuracy;
6. return the average of the counts found over all the iterations.

Now we formalize these intuitive steps, specifying in a rigorous way all the quantities involved in the final algorithm.

At *step* 1 the number of possible random timestamps, which corresponds to the number of possible graphs $\mathcal{T}_i$ to be gathered at iteration $i = 1, \ldots, s$ in *step* 2, is $\Delta_{\mathcal{T},1} = |\{e = (u, v, t) \in \mathcal{E} : t_1 \leq t \leq t_{last}\}|$ for t_{last} defined as in *step* 1. *Step* 3 employs an exact routine to output each motif $U \in \mathcal{U}$ that is contained in $\mathcal{T}_i$, for each motif the routine outputs the starting and ending time.

To specify the description from step 4 we need to define a first set of random variables $X_U^i, i = 1, \ldots, s, U \in \mathcal{U}$ where:

$$X_U^i = \begin{cases} 1 & \text{if motif } U \in \mathcal{U} \text{ is in the } i\text{-th sample } \mathcal{T}_i \text{ of the} \\ & \text{graph } \mathcal{T}; \\ 0 & \text{otherwise.} \end{cases}$$

Note that each X_U^i is a Bernoulli random variable; let us compute the probability of each of the X_U^i's to assume value 1. Let r_U be the number of possible random choices t_r from which we can count motif U gathering $\mathcal{T}_i$ at some iteration $i = 1, \ldots, s$, namely $r_U = |\{(u, v, t) \in \mathcal{E} : \max\{t_1, t_l^U - c\delta\} \leq t \leq \min\{t_{last}, t_1^U\}\}|$ for each motif instance $U \in \mathcal{U}$. Then:

$$\begin{aligned} \mathbb{P}(X_U^i = 1) = \mathbb{P}(&\mathcal{T}_i \text{ obtained from a random timestamp } t_r \\ &\text{at iteration } i \in \{1, \ldots, s\} \text{ contains motif } U) = \frac{r_U}{\Delta_{\mathcal{T},1}} = p_U \end{aligned} \tag{1}$$

Since each $X_U^i, i = 1, \ldots, s, U \in \mathcal{U}$ is a Bernoulli random variable then it also holds that $\mathbb{E}[X_U^i] = p_U$.

Now we can express the weighted count at *step* 4 for each iteration $i = 1, \ldots, s$ as a function of the variables already defined, this yields to defining the following random vector $\boldsymbol{X} = (X_1, \ldots, X_i, \ldots, X_s), i = 1, \ldots, s$ where:

$$X_i = \sum_{U \in \mathcal{U}} \frac{1}{p_U} X_U^i$$

Each variable X_i corresponds to the weighted count at iteration $i = 1, \ldots, s$ of the procedure at *step* 4, moreover observe that:

$$\mathbb{E}[X_i] = \mathbb{E}\left[\sum_{U \in \mathcal{U}} \frac{1}{p_U} X_U^i\right] \overset{\bigstar}{=} \sum_{U \in \mathcal{U}} \frac{1}{p_U} \mathbb{E}\left[X_U^i\right] \overset{(1)}{=} \sum_{U \in \mathcal{U}} \frac{1}{p_U} p_U = \sum_{U \in \mathcal{U}} 1 = C_M \quad (2)$$

where in $\bigstar$ we used the linearity of the expectation and in the next step we used equation (1). We showed that the expectation of each one of the X_i's for $i = 1, \ldots, s$ is exactly the quantity we want to estimate, moreover this result does not depend on the specific procedure as long as the weights accounts for the probability of each motif $U \in \mathcal{U}$ to be "sampled".

Now we can state and prove the following lemma which formalizes *step* 6:

Lemma 1. *$||\boldsymbol{X}||_1/s$ is an unbiased estimator for C_M.*

Proof. We have to prove that:

$$\mathbb{E}\left[\frac{||\boldsymbol{X}||_1}{s}\right] = C_M$$

this is done by considering the definition of $\boldsymbol{X}$, the linearity of expectation $(L.)$ and the equation (2), thus:

$$\mathbb{E}\left[\frac{||\boldsymbol{X}||_1}{s}\right] \overset{(L.)}{=} \frac{\mathbb{E}\left[||\boldsymbol{X}||_1\right]}{s} = \frac{1}{s}\mathbb{E}\left[\sum_{i=1}^{s} X_i\right] =$$
$$\overset{(L.)}{=} \frac{1}{s}\sum_{i=1}^{s} \mathbb{E}\left[X_i\right] \overset{(2)}{=} \frac{1}{s}\sum_{i=1}^{s} C_M = \frac{sC_M}{s} = C_M$$

□

Now we present the Algorithm 4 that formalizes the initial intuitive procedure, and then we prove that the output of Algorithm 4 is an ϵ-relative approximation to C_M with probability at least $1 - \eta$.

Algorithm 4: Temporal motif approximator - first variant

Input: $\mathcal{T} = (\mathcal{V}, \mathcal{E}), M = (\mathcal{K}, \sigma), \delta, \epsilon, \eta, c$
Output: $C_M^{'}$ such that $\mathbb{P}(|C_M^{'} - C_M| \geq \epsilon C_M) \leq \eta$

$m \leftarrow |\mathcal{E}|$
$t_{last} \leftarrow$ * Timestamp t of an edge of $\mathcal{T}$ that minimizes $|t - t_m + c\delta|$ among all timestamps and $t \geq t_m - c\delta$ *
$\Delta_{\mathcal{T},1} \leftarrow$ EDGE_COUNTER$(\mathcal{T}, t_1, t_{last})$
$s \leftarrow \left\lceil \frac{\Delta^2_{\mathcal{T},1}}{2\epsilon^2} \ln\left(\frac{2}{\eta}\right) \right\rceil$
$\boldsymbol{X} \leftarrow (X_1 = 0, \ldots, X_s = 0)$
for $i \leftarrow 1$ *to* s *(in parallel)* **do**
    $t_r \leftarrow$ RANDOM_TIMESTAMP$(\mathcal{T}, t_1, t_{last})$
    $\mathcal{T}_i \leftarrow$ TEMPORAL_GRAPH$(\mathcal{T}, t_r, t_r + c\delta)$
    $S \leftarrow$ EXACT_MOTIF_COUNTER$(\mathcal{T}_i, M, \delta)$
    foreach $(t_1^U, t_l^U) \in S$ **do**
        $r_U \leftarrow$ EDGE_COUNTER$(\mathcal{T}, \max\{t_1, t_l^U - c\delta\}, \min\{t_{last}, t_1^U\})$
        $p_U \leftarrow \frac{r_U}{\Delta_{\mathcal{T},1}}$
        $X_i \leftarrow X_i + \frac{1}{p_U}$
$C_M^{'} \leftarrow \frac{1}{s} \sum_{i=1}^{s} X_i$
return $C_M^{'}$

Where the extra routines used in Algorithm 4 are the following ones:

Algorithm 5: EXACT_MOTIF_COUNTER

Input: $\mathcal{T} = (\mathcal{V}, \mathcal{E}), M = (\mathcal{K}, \sigma), \delta$.
Output: The set $S = \{(t_1^U, t_l^U) : U \in \mathcal{U}\}$ of δ-instances of M in $\mathcal{T}$ with their respective starting time ad ending time.

/* Save in S for each instance U of M in the input graph the respective starting and ending time t_1^U, t_l^U */
return S

Algorithm 6: EDGE_COUNTER

Input: $\mathcal{T} = (\mathcal{V}, \mathcal{E}), a \in \mathbb{R}, b \in \mathbb{R}$ with $a \leq b$.
Output: Number of edges of $\mathcal{E}$ with a timestamp t such that $a \leq t \leq b$.

$m \leftarrow |\mathcal{E}|$
if $(b > t_m)$ **then**
    $b \leftarrow t_m$
return $r \leftarrow |\{(u, v, t) \in \mathcal{E} : a \leq t \leq b\}|$

Algorithm 7: RANDOM_TIMESTAMP

Input: $\mathcal{T} = (\mathcal{V}, \mathcal{E}), a \in \mathbb{R}, b \in \mathbb{R}$ with $a \leq b$.
Output: A timestamp t of an edge of $\mathcal{T}$ chosen at random such that $a \leq t \leq b$.
$(u, v, t) \leftarrow$ `Random edge of` $\mathcal{E}$ `chosen with uniform probability, such that` $a \leq t \leq b$
return t

Algorithm 8: TEMPORAL_GRAPH

Input: $\mathcal{T} = (\mathcal{V}, \mathcal{E}), t_a \in \mathbb{R}^+, t_b \in \mathbb{R}^+$ with $t_a \leq t_b$.
Output: The temporal graph with edges that have timestamps in $[t_a, t_b]$.
if $(t_b > t_m)$ **then**
  $t_b \leftarrow t_m$
$\mathcal{E}_G \leftarrow \{(u, v, t) \in \mathcal{E} : t_a \leq t \leq t_b\}$
$\mathcal{V}_G \leftarrow \{u \in \mathcal{V} : (u, v, t) \in \mathcal{E}_G \vee (v, u, t) \in \mathcal{E}_G\}$
return $(\mathcal{V}_G, \mathcal{E}_G)$

We already showed in lemma 1 that $C'_M = ||\boldsymbol{X}||_1/s$ is an unbiased estimator to C_M. In order to prove the correctness of the algorithm 4, we have to show that s is sufficiently large to achieve an ϵ-relative approximation to C_M with probability at least $1 - \eta$, for each $(\epsilon, \eta) \in (0, 1)^2$. To do this we need the following result from [11]:

Theorem (**Hoeffding bound 4.12**-[11]). *Let $X_1, \ldots, X_s$ be independent random variables such that for all $1 \leq i \leq s, \mathbb{E}[X_i] = \mu$ and $\mathbb{P}(a \leq X_i \leq b) = 1$. Then*

$$\mathbb{P}\left(\left|\frac{1}{s}\sum_{i=1}^{s} X_i - \mu\right| \geq \epsilon\right) \leq 2e^{-2s\epsilon^2/(b-a)^2}$$

Before applying this result we have to make some considerations on the random vector $\boldsymbol{X}$, in particular we have to limit the domain of it's components $X_i, i = 1, \ldots, s$:

1. first of all note that trivially $X_i \geq 0$ since in the worst case we do not count any motif $U \in \mathcal{U}$ at iteration $i \in \{1, \ldots, s\}$;

2. moreover note that:

$$X_i = \sum_{U \in \mathcal{U}} \frac{1}{p_U} X_U^i \overset{A.}{\leq} \sum_{U \in \mathcal{U}} \frac{1}{p_U} \overset{(1.)}{=} \sum_{U \in \mathcal{U}} \frac{\Delta_{\mathcal{T},1}}{r_U} =$$
$$= \Delta_{\mathcal{T},1} \sum_{U \in \mathcal{U}} \frac{1}{r_U} \overset{B.}{\leq} \Delta_{\mathcal{T},1} \sum_{U \in \mathcal{U}} 1 = \Delta_{\mathcal{T},1} C_M$$

where $A.$ accounts for the fact that each X_U^i assumes value in $\{0,1\}$ hence by setting all the $X_U^i = 1, U \in \mathcal{U}$ we are possibly adding positive terms to the sum. (1.) accounts for the definition of $p_U, U \in \mathcal{U}$. And $B.$ accounts for the fact that for each motif there exists at least one timestamp to be sampled that allows us to count that motif, otherwise the instance would not exist.

Thus 1 and 2 imply that $\mathbb{P}(0 \leq X_i \leq \Delta_{\mathcal{T},1} C_M) = 1, i = 1, \ldots, s$, this allows us to state the following lemma:

Lemma 2. *Given* $(\epsilon, \eta) \in (0,1)^2$ *let* $\boldsymbol{X} = (X_1, \ldots, X_s)$ *then if* $s \geq \frac{\Delta_{\mathcal{T},1}^2}{2\epsilon^2} \ln\left(\frac{2}{\eta}\right)$ *for algorithm 4 it holds that:*

$$\mathbb{P}\left(\left|\frac{1}{s}||\boldsymbol{X}||_1 - C_M\right| \geq \epsilon C_M\right) \leq \eta$$

Proof. We have to prove that for $s \geq \frac{\Delta_{\mathcal{T},1}^2}{2\epsilon^2} \ln\left(\frac{2}{\eta}\right)$ it holds:

$$\mathbb{P}\left(\left|\frac{1}{s}\sum_{i=1}^{s} X_i - C_M\right| \geq \epsilon C_M\right) \leq \eta$$

Recall that $\mathbb{E}[X_i] = C_M, \mathbb{P}(0 \leq X_i \leq \Delta_{\mathcal{T},1} C_M) = 1, i = 1, \ldots, s$ hence applying the Hoeffding bound $(H.)$ to the quantity of interest:

$$\mathbb{P}\left(\left|\frac{1}{s}\sum_{i=1}^{s} X_i - C_M\right| \geq \epsilon C_M\right) \overset{H.}{\leq}$$
$$\overset{H.}{\leq} 2e^{-2s(\epsilon C_M)^2/(\Delta_{\mathcal{T},1} C_M - 0)^2} = 2e^{-2s\epsilon^2/\Delta_{\mathcal{T},1}^2} \overset{I.}{\leq} \eta$$

Where $I.$ comes from the fact that we set $s \geq \frac{\Delta_{\mathcal{T},1}^2}{2\epsilon^2} \ln\left(\frac{2}{\eta}\right)$ which concludes the proof. □

3.2 Improved Algorithm

The first algorithm relied on the fact that, at each iteration we can sample a graph with timestamps in an interval of length $c\delta$ with $c > 1$, starting from some timestamp of an edge chosen randomly, this is not the only possibility, an alternative is presented in this section.

As first thing note that the timestamps of the edges of $\mathcal{T}$ are distributed in $[t_1, t_m]$ where $m = |\mathcal{E}|$, since we assume the timestamps to be sorted. Once the user specifies a motif M and a length δ, we can select a random number in $[t_l - c\delta, t_{m-l}]$, where l is the number of edges in M, as from Definition 2, and gather the edges of $\mathcal{T}$ with timestamps grater or equal than the random

Algorithm 9: Temporal motif approximator - second variant

Input: $\mathcal{T} = (\mathcal{V}, \mathcal{E}), M = (\mathcal{K}, \sigma), \delta, \epsilon, \eta, c$
Output: $C_M^{'}$ such that $\mathbb{P}(|C_M^{'} - C_M| \geq \epsilon C_M) \leq \eta$

1 $l \leftarrow |\mathcal{E}_{\mathcal{K}}|$; $m \leftarrow |\mathcal{E}|$
2 $\Delta_{\mathcal{T},2} \leftarrow t_{m-l} - t_l + c\delta$
3 $s \leftarrow \left\lceil \frac{\Delta_{\mathcal{T},2}^2}{2(c-1)^2\delta^2\epsilon^2} \ln\left(\frac{2}{\eta}\right) \right\rceil$
4 $\boldsymbol{X} \leftarrow (X_1 = 0, \ldots, X_s = 0)$
5 **for** $i \leftarrow 1$ *to* s *(in parallel)* **do**
6 $\quad t_r \leftarrow$ RANDOM_NUMBER$(t_l - c\delta, t_{m-l})$
7 $\quad \mathcal{T}_i \leftarrow$ TEMPORAL_GRAPH$(\mathcal{T}, t_r, t_r + c\delta)$
8 $\quad S \leftarrow$ EXACT_MOTIF_COUNTER$(\mathcal{T}_i, M, \delta)$
9 $\quad$ **foreach** $(t_1^U, t_l^U) \in S$ **do**
10 $\quad\quad \tilde{r}_U \leftarrow c\delta - (t_l^U - t_1^U)$
11 $\quad\quad \tilde{p}_U \leftarrow \frac{\tilde{r}_U}{\Delta_{\mathcal{T},2}}$
12 $\quad\quad X_i \leftarrow X_i + \frac{1}{\tilde{p}_U}$
13 $C_M^{'} \leftarrow \frac{1}{s}\sum_{i=1}^{s} X_i$
14 **return** $C_M^{'}$

number selected of at most $c\delta$ with $c > 1$.
Such idea leads to algorithm 9, which is quite similar to the algorithm 4 already presented, thus we can analyse it directly proving the correctness.

In line 2 we get $\Delta_{\mathcal{T},2} = t_{m-l} - t_l + c\delta$, the length of the interval $[t_l - c\delta, t_{m-l}]$; from this interval at each iteration $i = 1, \ldots, s$ we take a random number t_r in line 6.

In line 3 we set the number of iterations, which we discuss later. In line 4 we initialize the vector of counts for each iteration $i = 1, \ldots s$.

In line 7 we use algorithm 8 to gather the temporal graph $\mathcal{T}_i$ with edges $\{(u, v, t) : t_r \leq t \leq t_r + c\delta\}$, then we use the exact algorithm to get all the instances of motif M in $\mathcal{T}_i$, with length at most δ in line 8.

Let $X_U^i, i = 1, \ldots, s, U \in \mathcal{U}$ be defined as in the previous analysis. Let $\tilde{r}_U$ be the length of the interval from which a random number t_r from $[t_l - c\delta, t_{m-l}]$ allows to gather a graph $\mathcal{T}_i$ that contains motif $U \in \mathcal{U}$ at each iteration $i = 1, \ldots, s$ of algorithm 9 i.e., $\tilde{r}_U = c\delta - (t_l^U - t_1^U)$. Then $\forall U \in \mathcal{U}, i = 1, \ldots, s$:

$$\mathbb{P}(X_U^i = 1) = \mathbb{P}(\mathcal{T}_i \text{ obtained from a random number } t_r \text{ at iteration } i \in \{1, \ldots, s\} \text{ contains motif } U) = \frac{c\delta - (t_l^U - t_1^U)}{\Delta_{\mathcal{T},2}} = \frac{\tilde{r}_U}{\Delta_{\mathcal{T},2}} = \tilde{p}_U$$

Let $\boldsymbol{X} = (X_1, \ldots, X_i, \ldots, X_s)$ with each $X_i, i = 1, \ldots, s$ defined as in the previous analysis but substituting p_U with $\tilde{p}_U$, then the results (2) and

lemma 1 still hold, this reconciles with the fact that as long as the weight of each motif accounts for the probability of motif $U \in \mathcal{U}$ to be counted at iteration $i = 1, \dots, s$, the result is not constrained to the specific algorithm. The previous observations clarify the weighted count in lines 9-13.

We still have to prove that the number of iterations set in line 2 are sufficient to achieve the desired accuracy, to do so we present the following observations on the $X_i, i = 1, \dots, s$:

1. first of all note that trivially $X_i \geq 0$ since in the worst case we do not count any motif $U \in \mathcal{U}$ at iteration $i \in \{1, \dots, s\}$;

2. moreover note that:

$$X_i = \sum_{U \in \mathcal{U}} \frac{1}{\tilde{p}_U} X_U^i \overset{A.}{\leq} \sum_{U \in \mathcal{U}} \frac{1}{\tilde{p}_U} \overset{(1.)}{=} \sum_{U \in \mathcal{U}} \frac{\Delta_{\mathcal{T},2}}{\tilde{r}_U} =$$
$$= \Delta_{\mathcal{T},2} \sum_{U \in \mathcal{U}} \frac{1}{c\delta - (t_l^U - t_1^U)} \overset{B.}{\leq} \Delta_{\mathcal{T},2} \sum_{U \in \mathcal{U}} \frac{1}{c\delta - \delta} = \frac{\Delta_{\mathcal{T},2} C_M}{(c-1)\delta}$$

   where $A.$ accounts for the fact that each X_U^i assumes value in $\{0, 1\}$ hence by setting all the $X_U^i = 1, U \in \mathcal{U}$ we are possibly adding positive terms to the sum. (1.) uses the definition of $\tilde{p}_U, U \in \mathcal{U}$. $B.$ accounts for the fact that the quantity $t_l^U - t_1^U$ corresponds to the duration $\Delta(U)$ of motif $U \in \mathcal{U}$ hence by definition it holds $0 < \Delta(U) \leq \delta$.

Then 1 and 2 imply that $\mathbb{P}(0 \leq X_i \leq \Delta_{\mathcal{T},2} C_M / ((c-1)\delta)) = 1, i = 1, \dots, s$, this allows us to state the following lemma:

Lemma 3. *Given* $(\epsilon, \eta) \in (0,1)^2, c > 1$ *let* $\boldsymbol{X} = (X_1, \dots, X_s)$ *then if* $s \geq \frac{\Delta_{\mathcal{T},2}^2}{2(c-1)^2\delta^2\epsilon^2} \ln\left(\frac{2}{\eta}\right)$ *for algorithm 9 it holds that:*

$$\mathbb{P}\left(\left|\frac{1}{s}||\boldsymbol{X}||_1 - C_M\right| \geq \epsilon C_M\right) \leq \eta$$

Proof. We have to prove that for $s \geq \frac{\Delta_{\mathcal{T},2}^2}{2(c-1)^2\delta^2\epsilon^2} \ln\left(\frac{2}{\eta}\right)$ in algorithm 9 it holds:

$$\mathbb{P}\left(\left|\frac{1}{s}\sum_{i=1}^{s} X_i - C_M\right| \geq \epsilon C_M\right) \leq \eta$$

Recall that $\mathbb{E}[X_i] = C_M, \mathbb{P}(0 \leq X_i \leq \Delta_{\mathcal{T},2} C_M / ((c-1)\delta)) = 1, i = 1, \dots, s$ hence applying the Hoeffding bound $(H.)$ to the quantity of interest:

$$\mathbb{P}\left(\left|\frac{1}{s}\sum_{i=1}^{s} X_i - C_M\right| \geq \epsilon C_M\right) \overset{H.}{\leq}$$
$$\overset{H.}{\leq} 2e^{-2s(\epsilon C_M)^2 / (\Delta_{\mathcal{T},2} C_M / ((c-1)\delta) - 0)^2} = 2e^{-2s(c-1)^2\delta^2\epsilon^2/\Delta_{\mathcal{T},2}^2} \overset{I.}{\leq} \eta$$

Where $I.$ comes from the fact that we set $s \geq \frac{\Delta^2_{\mathcal{T},2}}{2(c-1)^2\delta^2\epsilon^2} \ln\left(\frac{2}{\eta}\right)$ which concludes the proof. □

The sample size we derived in lemma 3, is not the only possible choice for the parameter s, we observe that such choice has a crucial impact on the performances of our algorithm, thus to try to reduce such value we performed an analysis using the tool of martingales, such analysis is presented in the next section.

3.3 Limiting the sample size through martingales

Let us perform the analysis of the Algorithm 9 already presented using the tool of Martingales, we will present two main results:

- A first bound, similar to the bound already derived in lemma 3;
- An alternative bound, which may improve the size s, which is not computable thus may be useful only theoretically and not in practice.

We now give a short introduction to martingales, following the presentation from [11]. First of all a martingale is defined as follows.

Definition 9. A sequence of random variables $Z_0, Z_1, \ldots$ is a *martingale with respect to the sequence* $X_0, X_1, \ldots$ if, for all $n \geq 0$ the following conditions hold:

- Z_n is a function of $X_0, X_1, \ldots, X_n$;
- $\mathbb{E}[|Z_n|] < \infty$;
- $\mathbb{E}[Z_{n+1}|X_0, X_1, \ldots, X_n] = Z_n$.

A sequence of random variables $Z_0, Z_1, \ldots$ is called a *martingale* when it is a martingale with respect to itself. That is, $\mathbb{E}[|Z_n|] < \infty$, and $\mathbb{E}[Z_{n+1}|Z_0, \ldots, Z_n] = Z_n$.

To prove the bound on the sample size we will need the following result,

Theorem (**Azuma–Hoeffding Inequality 13.4** - [11]). *Let* $X_0, \ldots, X_n$ *be a martingale such that*

$$|X_k - X_{k-1}| \leq c_k.$$

Then, for all $t \geq 1$ *and any* $\lambda > 0$,

$$\mathbb{P}\left[|X_t - X_0| \geq \lambda\right] \leq 2e^{-\frac{\lambda^2}{2\sum_{k=1}^{t} c_k^2}}$$

3.3.1 A first bound

Let us consider the variables $X_1, \ldots, X_s$ already introduced, let $f(X_1, \ldots, X_s) = 1/s \sum X_i$ which is the function we use to obtain the estimate to our procedure. Let us define a Doob martingale (Chapter 11 of [11]) on the function f and the variables $X_1, \ldots, X_s$, in particular we define the following martingale:

- $Z_0 = \mathbb{E}[f(X_1, \ldots, X_s)] = \mathbb{E}\left[\frac{1}{s}\sum_{i=1}^{s} X_i\right] = C_M$;
- $Z_i = \mathbb{E}\left[\frac{1}{s}\sum_{j=1}^{s} X_j | X_1, \ldots X_i\right]$ with $i = 1, \ldots, s$, clearly Z_s is the value of the estimate at the end of the iterations of our algorithm, i.e., the final output C'_M.

As we have done in the previous analysis, we want to bound the probability $\mathbb{P}[|C'_M - C_M| \geq \epsilon C_M]$, to do so we need to analyse the martingale we already defined, in particular we need to bound the quantity $|Z_{i+1} - Z_i|, i = 0, \ldots, s-1$.

Let $i = 0$, then we have:

$$|Z_1 - Z_0| = \left|\mathbb{E}\left[\frac{1}{s}\sum_{j=1}^{s} X_j | X_1\right] - C_M\right| \overset{(1)}{=} \left|\frac{X_1}{s} + \left(\frac{1}{s}\sum_{j=2}^{s}\mathbb{E}[X_j]\right) - C_M\right| =$$
$$\overset{(2)}{=} \left|\frac{X_1}{s} + \left(\frac{1}{s}\sum_{j=2}^{s} C_M\right) - C_M\right| \overset{(3)}{=} \left|\frac{X_1}{s} + \frac{C_M(s-1)}{s} - C_M\right| =$$
$$\overset{(4)}{=} \left|\frac{X_1}{s} - \frac{C_M}{s}\right| \overset{(5)}{\leq} \left|\frac{\Delta_{\mathcal{T},2} C_M}{(c-1)\delta s} - \frac{C_M}{s}\right| \overset{(6)}{=} \frac{C_M}{s}\left(\frac{\Delta_{\mathcal{T},2}}{(c-1)\delta} - 1\right).$$

Where (1) comes from the linearity of expectation and the fact that the variables $X_j, j = 2, \ldots, s$ are independent from the variable X_1, (2) uses the equality $\mathbb{E}[X_j] = C_M, j = 1, \ldots, s$, (3) and (4) are just a rearrangement of the terms. (5) uses the bound on the domain of X_1, while (6) it holds since the term is positive so we can remove the absolute value.

Let $i \in \{1, \ldots, s-1\}$, then:

$$|Z_{i+1} - Z_i| \overset{(1)}{=} \left|\mathbb{E}\left[\frac{1}{s}\sum_{j=1}^{s} X_j | X_1, \ldots, X_{i+1}\right] - \mathbb{E}\left[\frac{1}{s}\sum_{j=1}^{s} X_j | X_1, \ldots, X_i\right]\right|$$
$$\overset{(2)}{=} \left|\frac{X_1}{s} + \cdots + \frac{X_{i+1}}{s} + \left(\frac{1}{s}\sum_{j=i+2}^{s}\mathbb{E}[X_j]\right) - \frac{X_1}{s} - \cdots - \frac{X_i}{s} -\right.$$
$$\left. - \left(\frac{1}{s}\sum_{j=i+1}^{s}\mathbb{E}[X_j]\right)\right| \overset{(3)}{\leq} \left|\frac{\Delta_{\mathcal{T},2} C_M}{(c-1)\delta s} - \frac{C_M}{s}\right| = \frac{C_M}{s}\left(\frac{\Delta_{\mathcal{T},2}}{(c-1)\delta} - 1\right).$$

In (1) we used the definition of the variables $Z_i, i = 1, \dots, s$, while in (2) we used the linearity of the expectation and the independence of the variables $X_i, i = 1, \dots, s$, we observe that if the lower index of the summation is greater than the upper one we are referring to the empty sum. Step (3) comes from the fact that $X_j, j = 1, \dots, i$ have the same values and opposite signs, thus may be simplified and the fact that $\mathbb{E}[X_j] = C_M, j = 1, \dots, s$. The last step is just the same as the previous case.

Lemma 4. *Given $(\epsilon, \eta) \in (0,1)^2, c > 1$ let $X_1, \dots, X_s$ be defined as in the algorithm 9, then if $s \geq \frac{2}{\epsilon^2}\left(\frac{\Delta_{\mathcal{T},2}}{(c-1)\delta} - 1\right)^2 \log\left(\frac{2}{\eta}\right)$ for algorithm 9 it holds that:*

$$\mathbb{P}\left(\left|C'_M - C_M\right| \geq \epsilon C_M\right) \leq \eta$$

Proof. Let $Z_i, i = 0, \dots, s$ be the martingale defined above, we have to prove that for $s \geq \frac{2}{\epsilon^2}\left(\frac{\Delta_{\mathcal{T},2}}{(c-1)\delta} - 1\right)^2 \log\left(\frac{2}{\eta}\right)$ in algorithm 9 it holds:

$$\mathbb{P}\left(\left|C'_M - C_M\right| \geq \epsilon C_M\right) \leq \eta = \mathbb{P}\left(|Z_t - Z_0| \geq \epsilon C_M\right) \leq \eta$$

We already showed that $\forall k = 1, \dots, s$ it holds

$$|Z_k - Z_{k-1}| \leq \frac{C_M}{s}\left(\frac{\Delta_{\mathcal{T},2}}{(c-1)\delta} - 1\right).$$

Thus applying the Azuma-Hoeffding Inequality in step (1.) and the fact that we chose $s \geq \frac{2}{\epsilon^2}\left(\frac{\Delta_{\mathcal{T},2}}{(c-1)\delta} - 1\right)^2 \log\left(\frac{2}{\eta}\right)$ (2.) we obtain:

$$\begin{aligned}\mathbb{P}\left(|Z_s - Z_0| \geq \epsilon C_M\right) &\overset{(1.)}{\leq} 2e^{-\frac{\epsilon^2 C_M^2}{2\frac{C_M^2}{s}\left(\frac{\Delta_{\mathcal{T},2}}{(c-1)\delta}-1\right)^2}} \\ &= 2e^{-\frac{\epsilon^2 s}{2\left(\frac{\Delta_{\mathcal{T},2}}{(c-1)\delta}-1\right)^2}} \overset{(2.)}{\leq} \eta.\end{aligned}$$

□

Observe that such bound has the same order of magnitude as the one already derived, thus it may be not efficient to use such bound in practice, in the next section we devise a new bound which may be significantly better than the ones already presented.

3.3.2 An alternative bound

In this section we try to derive a different bound to the number s of iteration to achieve the desired (ϵ, η)-approximation of algorithm 9; the core idea is to unpack the variables $X_i, i = 1, \dots, s$ and exploit the dependencies of the variables $X_U^i, U \in \mathcal{U}$, $i = 1, \dots, s$. Suppose w.l.o.g., we labelled the

motifs instances as $U_1, \dots, U_{C_M}$, we want to rewrite the process as function of the variables $X_U^j, U \in \mathcal{U}, j = 1, \dots, s$ which are exactly sC_M, thus we define,

$$\hat{X}_i = \begin{cases} 1 & \text{if motif } U_{i-\left(\lceil \frac{i}{C_M} \rceil - 1\right)C_M} \text{ is in the sample } \lceil \frac{i}{C_M} \rceil \text{ of } \mathcal{T},\ i = 1, \dots, sC_M; \\ 0 & \text{otherwise.} \end{cases}$$

let us denote $f(i) = i - \left(\lceil \frac{i}{C_M} \rceil - 1\right) C_M$ and $g(i) = \lceil \frac{i}{C_M} \rceil$ then we have,

$$\mathbb{P}(\hat{X}_i = 1) = \mathbb{P}(X^{g(i)}_{U_{f(i)}} = 1) = \tilde{p}_{U_{f(i)}}, i = 1, \dots, sC_M$$

were with a slightly abuse of notation (since now the motifs are labelled) we referred to the same variables $X_U^j, j = 1, \dots, s, U \in \mathcal{U}$ used in the definition of $X_j, j = 1, \dots, s$, thus $\tilde{p}_{U_{f(i)}} = \tilde{p}_U$ if $U = U_{f(i)}$. Based on these definition the estimator used in the algorithm is the following,

$$\frac{1}{s} \sum_{i=1}^{sC_M} \frac{1}{\tilde{p}_{U_{f(i)}}} \hat{X}_i.$$

Then we may rephrase the algorithm 9 as the following stochastic process:

- Suppose we have s random timestamps t_r, at each step $i = 1, \dots, sC_M$ we are given the value of $\hat{X}_i$, which corresponds to the information "the instance $U_{f(i)}$ is/is not contained in the sample generated from the timestamp number $\lceil \frac{i}{C_M} \rceil$".

- At each step $i = 1, \dots, sC_M$ we sum $\frac{1}{s\tilde{p}_{U_{f(i)}}}$ to our estimate if $\hat{X}_i = 1$.

Such process is a rephrasing of the Algorithm 9 which suggests a very intuitive way to define a Doob Martingale on the variables already defined, in particular:

- $Z_0 = \mathbb{E}[f(\hat{X}_1, \dots, \hat{X}_{sC_M})] = \mathbb{E}\left[\frac{1}{s} \sum_{i=1}^{sC_M} \frac{1}{\tilde{p}_{U_{f(i)}}} \hat{X}_i\right] = C_M$;

- $Z_i = \mathbb{E}\left[\frac{1}{s} \sum_{i=1}^{sC_M} \frac{1}{\tilde{p}_{U_{f(i)}}} \hat{X}_i | \hat{X}_1, \dots, \hat{X}_i\right]$ with $i = 1, \dots, sC_M$, clearly as in the previous analysis Z_{sC_M} is the value of the estimate at the end of the iterations of our algorithm, i.e., the final output C'_M.

To apply the Azuma-Hoeffding inequality we need to bound the quantity $|Z_{i+1} - Z_i|, i = 0, \ldots, sC_M - 1$, thus:

$$
\begin{aligned}
|Z_{i+1} - Z_i| &\overset{(1)}{=} \left| \mathbb{E}\left[\frac{1}{s} \sum_{j=1}^{sC_M} \frac{1}{\tilde{p}_{U_{f(j)}}} \hat{X}_j | \hat{X}_1, \ldots, \hat{X}_{i+1} \right] - \right. \\
&\quad \left. - \mathbb{E}\left[\frac{1}{s} \sum_{j=1}^{sC_M} \frac{1}{\tilde{p}_{U_{f(j)}}} \hat{X}_j | \hat{X}_1, \ldots, \hat{X}_i \right] \right| \\
&\overset{(l.)}{=} \left| \left[\frac{1}{s} \sum_{j=1}^{sC_M} \frac{1}{\tilde{p}_{U_{f(j)}}} \mathbb{E}[\hat{X}_j | \hat{X}_1, \ldots, \hat{X}_{i+1}] \right] - \left[\frac{1}{s} \sum_{j=1}^{sC_M} \frac{1}{\tilde{p}_{U_{f(j)}}} \mathbb{E}[\hat{X}_j | \hat{X}_1, \ldots, \hat{X}_i] \right] \right| \\
&\overset{(2)}{=} \left| \frac{X_1}{s\tilde{p}_{U_{f(1)}}} + \ldots + \frac{X_i}{s\tilde{p}_{U_{f(i)}}} + \frac{X_{i+1}}{s\tilde{p}_{U_{f(i+1)}}} + \left[\frac{1}{s} \sum_{j=i+2}^{sC_M} \frac{1}{\tilde{p}_{U_{f(j)}}} \mathbb{E}[\hat{X}_j | \hat{X}_1, \ldots, \hat{X}_{i+1}] \right] - \right. \\
&\quad \left. - \frac{X_1}{s\tilde{p}_{U_{f(1)}}} - \cdots - \frac{X_i}{s\tilde{p}_{U_{f(i)}}} - \left[\frac{1}{s} \sum_{j=i+1}^{sC_M} \frac{1}{\tilde{p}_{U_{f(j)}}} \mathbb{E}[\hat{X}_j | \hat{X}_1, \ldots, \hat{X}_i] \right] \right| \\
&\overset{(3)}{=} \left| \frac{X_{i+1}}{s\tilde{p}_{U_{f(i+1)}}} + \left[\frac{1}{s} \sum_{j=i+2}^{sC_M} \frac{1}{\tilde{p}_{U_{f(j)}}} \mathbb{E}[\hat{X}_j | \hat{X}_1, \ldots, \hat{X}_{i+1}] \right] \right. \\
&\quad \left. - \left[\frac{1}{s} \sum_{j=i+1}^{sC_M} \frac{1}{\tilde{p}_{U_{f(j)}}} \mathbb{E}[\hat{X}_j | \hat{X}_1, \ldots, \hat{X}_i] \right] \right|
\end{aligned}
$$

Where (1) is from the definition of Z_{i+1} and Z_i, $(l.)$ is from the linearity of expectation, (2) is from the fact that in Z_{i+1} and Z_i respectively the first $i+1$ and the first i variables are given and (3) from the fact that the first i variables have the same values with opposite signs. Observe that now the variables may be dependent thus we cannot remove the conditional expectation. Let us denote the last equation with the symbol (★) for brevity.

In order to bound (★), we need to distinguish the following cases:

1. A : $\hat{X}_i$ and $\hat{X}_{i+1}$ belong to the same sample, thus $\lceil \frac{i}{C_M} \rceil = \lceil \frac{i+1}{C_M} \rceil$;
2. B: $\hat{X}_i$ and $\hat{X}_{i+1}$ belong to different samples, thus $\lceil \frac{i}{C_M} \rceil + 1 = \lceil \frac{i+1}{C_M} \rceil$.

Let now consider the Case A, using the fact that by construction the variables $\hat{X}_i, i = 1, \ldots, sC_M$ are dependent only if they belong to same sample we may rewrite the sums as follows,

$$
\begin{aligned}
(\bigstar) &\overset{(1)}{=} \left| \frac{X_{i+1}}{s\tilde{p}_{U_{f(i+1)}}} + \left[\frac{1}{s} \sum_{j=i+2}^{g(i+1)C_M} \frac{1}{\tilde{p}_{U_{f(j)}}} \mathbb{E}[\hat{X}_j | \hat{X}_1, \ldots, \hat{X}_{i+1}] \right] \right. \\
&+ \left[\frac{1}{s} \sum_{j=g(i+1)C_M+1}^{sC_M} \frac{1}{\tilde{p}_{U_{f(j)}}} \mathbb{E}[\hat{X}_j] \right] - \left[\frac{1}{s} \sum_{j=i+1}^{g(i)C_M} \frac{1}{\tilde{p}_{U_{f(j)}}} \mathbb{E}[\hat{X}_j | \hat{X}_1, \ldots, \hat{X}_i] \right] \\
&\left. - \left[\frac{1}{s} \sum_{j=g(i)C_M+1}^{sC_M} \frac{1}{\tilde{p}_{U_{f(j)}}} \mathbb{E}[\hat{X}_j] \right] \right| = \\
&\overset{(2)}{=} \left| \frac{X_{i+1}}{s\tilde{p}_{U_{f(i+1)}}} + \left[\frac{1}{s} \sum_{j=i+2}^{g(i+1)C_M} \frac{1}{\tilde{p}_{U_{f(j)}}} \mathbb{E}[\hat{X}_j | \hat{X}_1, \ldots, \hat{X}_{i+1}] \right] \right. \\
&\left. - \left[\frac{1}{s} \sum_{j=i+1}^{g(i+1)C_M} \frac{1}{\tilde{p}_{U_{f(j)}}} \mathbb{E}[\hat{X}_j | \hat{X}_1, \ldots, \hat{X}_i] \right] \right|
\end{aligned}
$$

In (1) we split each sum of ($\bigstar$) in two terms, the first is the sum on the current sample, the second term is the sum on the other samples, and we used the fact that only the variables in the current sample are dependent of the variables $\hat{X}_1, \ldots, \hat{X}_i$. In (2) we used the fact that we are in the case where $g(i) = g(i+1)$ since we assumed $\hat{X}_i, \hat{X}_{i+1}$ to be in the same sample. Observe that if the lower index of the summation is greater than the upper one we denote the sum as the empty sum. Let us denote the last equation with the symbol ($\bigstar\bigstar$) for brevity. In order to bound ($\bigstar\bigstar$), let $a = (g(i)-1)C_M + 1$, which is the first index of the variable in the current sample, then clearly the sum on the variables on the current sample is dependent only to the variables from $X_a, X_{a+1}, \ldots$, then we distinguish the following cases:

1. There exists in $\hat{X}_a, \ldots, \hat{X}_i$ at least one variable that assumes value 1;
2. No variables in $\hat{X}_a, \ldots, \hat{X}_i$ has value 1.

Case 1., let $a \leq i_1 < \cdots < i_q \leq i$ be the indexes of the variables that assume value 1, we define $\mathcal{U}_{U_j} = \{U \in \mathcal{U} : (t_1^U \geq t_l^{U_j} - c\delta) \wedge (t_l^U \leq t_1^{U_j} + c\delta) \wedge U \neq U_j\}$, then the only motifs at time i that have the possibility to be accounted for, are the motifs in $\mathcal{U}_{U_{i_1}} \cap \cdots \cap \mathcal{U}_{U_{i_q}}$, and their probability to be in the sample may be computed, step which we will avoid. Now three sub-cases may arise,

- I : the motif instance associated with the random variable $\hat{X}_{i+1}$ is in the set $\mathcal{U}_{U_{i_1}} \cap \cdots \cap \mathcal{U}_{U_{i_q}}$ at time i, and $\hat{X}_{i+1} = 1$;
- II : same situation as the case I but now $\hat{X}_{i+1} = 0$;
- III : the motif instance associated with the random variable $\hat{X}_{i+1}$ is not in the set $\mathcal{U}_{U_{i_1}} \cap \cdots \cap \mathcal{U}_{U_{i_q}}$, thus $\hat{X}_{i+1} = 0$.

In case I we have that $|\mathcal{U}_{U_{i_1}} \cap \cdots \cap \mathcal{U}_{U_{i_q}}| \geq |\mathcal{U}_{U_{i_1}} \cap \cdots \cap \mathcal{U}_{U_{i_q}} \cap \mathcal{U}_{U_{f(i+1)}}|$ since they are respectively at time i and at time $i+1$ the sets of the only possible motif instance that may have been sampled in the current sample. The maximum difference in this case, arises when, at time i all the motifs in $\mathcal{U}_{U_{i_1}} \cap \cdots \cap \mathcal{U}_{U_{i_q}}$ have a starting time greater or equal and an ending time lower or equal than the motif corresponding to variable $\hat{X}_{i+1}$, thus in this case,

$$
\begin{aligned}
(\bigstar\bigstar) &\overset{(1.)}{\leq} \left| \frac{\Delta_{\mathcal{T},2}}{s(c-1)\delta} + \left[\frac{\Delta_{\mathcal{T},2}}{s(c-1)\delta} \sum_{U_j \in \mathcal{U}_{U_{i_1}} \cap \cdots \cap \mathcal{U}_{U_{i_u}} \cap \mathcal{U}_{U_{f(i+1)}}} 1 \right] \right. \\
&\left. - \left[\frac{\alpha_1 \alpha_2 \Delta_{\mathcal{T},2}}{s(c-1)\delta} \sum_{U_j \in \mathcal{U}_{U_{i_1}} \cap \cdots \cap \mathcal{U}_{U_{i_q}}} 1 \right] \right| \\
&\overset{(2.)}{\leq} \left| \frac{\Delta_{\mathcal{T},2}}{s(c-1)\delta} + \frac{\Delta_{\mathcal{T},2}}{s(c-1)\delta}(\breve{C}^*_M - f(i) - 1) - \frac{\alpha_1 \alpha_2 \Delta_{\mathcal{T},2}}{s(c-1)\delta}(\breve{C}^*_M - f(i)) \right| \\
&\overset{(3.)}{\leq} \frac{\Delta_{\mathcal{T},2}}{s(c-1)\delta}((\breve{C}^*_M - 1) - \alpha_1 \alpha_2 (\breve{C}^*_M - 1)) = \underbrace{\frac{\Delta_{\mathcal{T},2}}{s(c-1)\delta}(\breve{C}^*_M - 1)(1 - \alpha_1 \alpha_2)}_{\Gamma_1/s}
\end{aligned}
$$

Where in (1.) we used the fact that $X_{i+1} = 1$ and the consideration made above, in particular if all the motifs instances have a starting time greater or equal and an ending time lower or equal than the motif corresponding to variable $\hat{X}_{i+1}$, then their probability to be in the sampled, given that $\hat{X}_{i+1} = 1$, is just 1 and the well known bound on each $1/\tilde{p}_U$. In the negative term we introduced two "constants" $0 < \alpha_1, \alpha_2 < 1$ which account for the fact that the expectation of all the motifs in $\mathcal{U}_{U_{i_1}} \cap \cdots \cap \mathcal{U}_{U_{i_q}}$ at time i may be lower than 1, thus α_1 is the minimum conditional expectation of a motif in such set; and α_2 accounts for the fact that the bound on each $1/\tilde{p}_U$ generally it is lower than the value used. In step (2.) we bounded the previous term introducing $\breve{C}^*_M = \max_{U \in \mathcal{U}}\{|\mathcal{U}_U|\}$ which maximizes the positive term, we introduced also $\breve{C}^*_M$ in the negative term since both positive and negative terms are dependent, recall that $|\mathcal{U}_{U_{i_1}} \cap \cdots \cap \mathcal{U}_{U_{i_q}}| \geq |\mathcal{U}_{U_{i_1}} \cap \cdots \cap \mathcal{U}_{U_{i_q}} \cap \mathcal{U}_{U_{f(i+1)}}|$. In the step (3.) we used the fact that for $f(i) \geq 1$ and we maximized the positive term. We observe that such final it may be lossy and it is computationally heavy to be estimated since all the motifs instances have to be identified, which is the problem we want address, in particular a more accurate analysis may be needed to understand better how to estimate α_1, α_2.

Case II. In such situation the worst case is similar to the previous one, all the motifs have a starting time greater or equal and an ending time lower

or equal than the motif corresponding to variable $\hat{X}_{i+1}$ then,

$$(\bigstar\bigstar) \overset{(1.)}{\leq} \left| - \left[\frac{1}{s} \sum_{U_j \in \mathcal{U}_{U_{i_1}} \cap \cdots \cap \mathcal{U}_{U_{i_q}}} \frac{1}{\tilde{p}_{U_j}} \mathbb{P}[\hat{X}_j = 1 | \hat{X}_a, \ldots, \hat{X}_{i+1}] \right] \right| \overset{(2.)}{\leq} \underbrace{\frac{\Delta_{\mathcal{T},2}}{s(c-1)\delta} \breve{C}^*_M}_{\Gamma_2/s}$$

In the first inequality (1.) we used the event where all the motif instances are "contained" in the motif corresponding to the variable $\hat{X}_{i+1}$, thus they are not in the sample since such variable has value 0. In the last step we applied the absolute value and introduced $\breve{C}^*_M$ which is defined as above, which clearly bounds the quantity of interest.

Case III. In such case it is easy to verify that the quantity we want to bound is 0 since we already had the information that such was not in the sample so the expectation at time $i+1$ is the same as the expectation at time i.

Let us look at case 2 now. In this case no variable among $\hat{X}_a, \ldots, \hat{X}_i$ has value 1 at time i, thus in this case it holds,

$$\begin{aligned}(\bigstar\bigstar) &\overset{(1)}{\leq} \left| \frac{\Delta_{\mathcal{T},2}}{s(c-1)\delta} + \left[\frac{\Delta_{\mathcal{T},2}}{s(c-1)\delta} \sum_{U_j \in \mathcal{U}_{U_{f(i+1)}}} 1 \right] - \right. \\ &\left. - \left[\frac{1}{s} \sum_{j=i+1}^{\lceil \frac{i+1}{C_M} \rceil C_M} \frac{1}{\tilde{p}_{U_{f(j)}}} \mathbb{P}[\hat{X}_j = 1 | \hat{X}_a = 0, \ldots, \hat{X}_i = 0] \right] \right| \\ &\overset{(2)}{\leq} \left| \frac{\Delta_{\mathcal{T},2}}{s(c-1)\delta} (\breve{C}^*_M + 1) - \frac{\beta_1 \beta_2 \Delta_{\mathcal{T},2}}{s(c-1)\delta} (\breve{C}^*_M + 1) \right| \\ &= \underbrace{\frac{\Delta_{\mathcal{T},2}}{s(c-1)\delta} (\breve{C}^*_M + 1)(1 - \beta_1\beta_2)}_{\Gamma_3/s}\end{aligned}$$

Where in (1) we used the fact that $\hat{X}_{i+1} = 1$, we used the bound on $1/\tilde{p}_{U_{f(j)}}$ and we bounded $\mathbb{E}[\hat{X}_j | \hat{X}_1, \ldots, \hat{X}_{i+1}]$ with one since it is a Bernoulli r.v.. In step (2) we used the bound on $|\mathcal{U}_U|, \forall U \in \mathcal{U}$ and the fact that at time i still holds that the number of instances over which the expectation is computed is greater or equal than the one at step at $i+1$; observe that also in this case we introduced $0 < \beta_1, \beta_2 < 1$ which account respectively for the minimum $\mathbb{P}[\hat{X}_j = 1 | \hat{X}_a = 0, \ldots, \hat{X}_i = 0]$ and the minimum $1/\tilde{p}_{U_{f(j)}}$ at step i.

Now it remains to analyse only the Case B, where in the original quantity $\hat{X}_{i+1}$ and $\hat{X}_i$ are from different samples, then in this case,

$$
\begin{aligned}
(\bigstar) &\overset{(1)}{\leq} \left| \frac{\Delta_{\mathcal{T},2}}{s(c-1)\delta} + \left[\frac{\Delta_{\mathcal{T},2}}{s(c-1)\delta} \sum_{U_j \in \mathcal{U}_{U_{f(i+1)}}} \mathbb{P}[\hat{X}_{U_j} = 1 | \hat{X}_{i+1} = 1] \right] + \right. \\
&\left. + \left(\frac{s - \lceil \frac{i+1}{C_M} \rceil}{s} \right) C_M - \left(\frac{s - \lceil \frac{i+1}{C_M} \rceil + 1}{s} \right) C_M \right| \\
&\overset{(2)}{\leq} \underbrace{\left| \frac{\Delta_{\mathcal{T},2}}{s(c-1)\delta} (\hat{C}^*_M + 1) - \frac{C_M}{s} \right|}_{\Gamma_4/s}
\end{aligned}
$$

Where in (1) we set $\hat{X}_{i+1} = 1$ and we bound it's probability, then the positive term is break in two sums the first on the current sample for which we conditioned and the second independent of the variable $\hat{X}_{i+1}$ on the next samples (if there exist), instead the negative term is the expectation from sample $\lceil \frac{i+1}{C_M} \rceil$ to the last sample without any condition. The last step (2) uses the bound on the positive sum as by definition of $\hat{C}^*_M$, which is reported in the in section 3.4.2, and the fact that $\frac{(s-g(i+1))}{s} C_M - \frac{(s-g(i+1)+1)}{s} C_M$ it is just $-\frac{C_M}{s}$.

Thus we just proved that for $i = 0, \ldots, sC_M - 1$ it holds,

$$|Z_{i+1} - Z_i| \leq \frac{\max\{\Gamma_1, \Gamma_2, \Gamma_3, \Gamma_4\}}{s} = \frac{\Gamma^*}{s}$$

Then we can state the following lemma.

Lemma 5. *Given $(\epsilon, \eta) \in (0,1)^2, c > 1$ let $\hat{X}_1, \ldots, \hat{X}_{sC_M}$ be the random variables defined in this section, then if $s \geq \frac{2(\Gamma^*)^2}{\epsilon^2 C_M} \left(\frac{2}{\eta} \right)$ for algorithm 9 it holds that:*

$$\mathbb{P}\left(\left| C'_M - C_M \right| \geq \epsilon C_M \right) \leq \eta$$

Proof. Let $Z_i, i = 0, \ldots, sC_M$ be the martingale defined above, we have to prove that for $s \geq \frac{2(\Gamma^*)^2}{\epsilon^2 C_M} \left(\frac{2}{\eta} \right)$ in algorithm 9 it holds:

$$\mathbb{P}\left(\left| C'_M - C_M \right| \geq \epsilon C_M \right) \leq \eta = \mathbb{P}\left(|Z_{sC_M} - Z_0| \geq \epsilon C_M \right) \leq \eta$$

We already showed that $\forall k = 1, \ldots, sC_M$ it holds

$$|Z_k - Z_{k-1}| \leq \frac{\Gamma^*}{s}.$$

Thus applying the Azuma-Hoeffding Inequality in step (1.) and the fact that we chose $s \geq \frac{2(\Gamma^*)^2}{\epsilon^2 C_M} \left(\frac{2}{\eta} \right)$ (2.) we obtain:

$$
\begin{aligned}
\mathbb{P}\left(|Z_s - Z_0| \geq \epsilon C_M \right) &\overset{(1.)}{\leq} 2e^{-\frac{\epsilon^2 C_M^2}{2sC_M \frac{(\Gamma^*)^2}{s^2}}} \\
&= 2e^{-\frac{\epsilon^2 sC_M}{2(\Gamma^*)^2}} \overset{(2.)}{\leq} \eta.
\end{aligned}
$$

which concludes the proof. □

As already mentioned computing such bound is much more difficult and computationally heavy than estimating exactly the number of δ-instances of a motif in a temporal network. This is also due to the fact that no tight and efficiently computable upper bounds exist for the quantities involved in the computation of Γ^*. Thus unfortunately we cannot evaluate such bound in practice.

3.4 Variance analysis

In this section we want to bound the variance of our estimator, in particular we want to bound the variance of the estimator used in algorithm 9. We will develop two analysis, the first one follows the idea of [9] while the second is completely new and may be significantly tighter.

3.4.1 A first upper bound

We recall that our estimator is:

$$\frac{1}{s}||\boldsymbol{X}||_1 = \frac{1}{s}\sum_{i=1}^{s} X_i.$$

Lemma 6. *In algorithm 9 it holds that,* $\mathrm{Var}\left(\frac{1}{s}||\boldsymbol{X}||_1\right) = \mathrm{Var}\left(\frac{1}{s}\sum_{i=1}^{s} X_i\right) \leq \frac{C_M^2}{s}\left(\frac{\Delta_{\mathcal{T},2}}{(c-1)\delta} - 1\right)$

Proof. We start by observing that,

$$\mathrm{Var}\left(\frac{1}{s}\sum_{i=1}^{s} X_i\right) = \frac{1}{s^2}\sum_{i=1}^{s}\mathrm{Var}(X_i)$$

since the variables $X_i, i = 1, \ldots, s$ are independent and by the property $\mathrm{Var}(aX) = a^2\mathrm{Var}(X)$. In order to evaluate the variance of the estimator we need to bound the variance of the variables $X_i, i = 1, \ldots, s$. We observe that $\mathrm{Var}(X_i) = \mathbb{E}[X_i^2] - \mathbb{E}[X_i]^2, i = 1, \ldots, s$ and we recall that $\mathbb{E}[X_i] = C_M$, thus we need to bound the quantity $\mathbb{E}[X_i^2]$,

$$\mathbb{E}[X_i^2] = \mathbb{E}\left[\left(\sum_{U\in\mathcal{U}}\frac{1}{\tilde{p}_U}X_U^i\right)^2\right] \overset{(1)}{=} \mathbb{E}\left[\sum_{U_1\in\mathcal{U}}\sum_{U_2\in\mathcal{U}}\frac{1}{\tilde{p}_{U_1}\tilde{p}_{U_2}}X_{U_1}^i X_{U_2}^i\right] =$$

$$\overset{(l.)}{=} \sum_{U_1\in\mathcal{U}}\sum_{U_2\in\mathcal{U}}\frac{1}{\tilde{p}_{U_1}\tilde{p}_{U_2}}\mathbb{E}[X_{U_1}^i X_{U_2}^i] \overset{(2)}{\leq} \sum_{U_1\in\mathcal{U}}\sum_{U_2\in\mathcal{U}}\frac{1}{\tilde{p}_{U_1}\tilde{p}_{U_2}}\mathbb{E}[X_{U_1}^i] =$$

$$\overset{(3)}{=} \sum_{U_1\in\mathcal{U}}\sum_{U_2\in\mathcal{U}}\frac{1}{\tilde{p}_{U_2}} \overset{(4)}{\leq} \sum_{U_1\in\mathcal{U}}\sum_{U_2\in\mathcal{U}}\frac{\Delta_{\mathcal{T},2}}{(c-1)\delta} = C_M^2\frac{\Delta_{\mathcal{T},2}}{(c-1)\delta}$$

Where (1) comes from the property of sums $(\sum_i a_i)^2 = (a_1 + \cdots + a_n) \cdot (a_1 + \cdots + a_n) = (a_1a_1 + \cdots + a_1a_n + a_2a_1 + \cdots + a_na_n) = \sum_i \sum_j a_ia_j$, then we apply the linearity of expectation ($l.$). In (2) we apply the inequality $\mathbb{E}[X^i_{U_1}X^i_{U_2}] \leq \mathbb{E}[X^i_{U_1}]$ and in step (3) we use the fact that $\mathbb{E}[X^i_{U_1}] = \tilde{p}_{U_1}$ from the definition of such variable, in (4) we use the bound on the value of $\tilde{p}_{U_2}$. The last equality comes from the fact that the two summations range over the set of the motifs instances, and we are summing 1 for each instance, since the inner term may be collected out of the summations due to it's independency of the index of the two sums. Based on the inequalities we derived we can bound the variance of the variables $X_i, i = 1, \ldots, s$ in particular,

$$\mathrm{Var}(X_i) = \mathbb{E}[X_i^2] - \mathbb{E}[X_i]^2 \leq C_M^2 \frac{\Delta_{\mathcal{T},2}}{(c-1)\delta} - C_M^2 = C_M^2 \left(\frac{\Delta_{\mathcal{T},2}}{(c-1)\delta} - 1 \right).$$

Note that such bound does not depend on the index $i = 1, \ldots, s$, thus substituting in the original summation we obtain,

$$\mathrm{Var}\left(\frac{1}{s} \sum_{i=1}^{s} X_i \right) \leq \frac{C_M^2}{s} \left(\frac{\Delta_{\mathcal{T},2}}{(c-1)\delta} - 1 \right).$$

□

3.4.2 An improved upper bound

In this section we want to perform another variance analysis, which may better give the intuition behind the estimate we are considering. We highlight that in the worst case, the bound we are going to present is similar to the one already derived, but in many cases it may be significantly better.

The idea is to use the same arguments of the bound already derived until the application of the inequality $\mathbb{E}[X^i_{U_1}X^i_{U_2}] \leq \mathbb{E}[X^i_{U_1}]$ (thus until step (2) of the previous proof) which we want to estimate in a different way, that is we want to understand better the value of $\mathbb{E}[X^i_{U_1}X^i_{U_2}]$. We observe that:

$$\begin{aligned}\mathbb{E}[X^i_{U_1}X^i_{U_2}] &= 1 \cdot \mathbb{P}(X^i_{U_1} = 1 \wedge X^i_{U_2} = 1) = \mathbb{P}(X^i_{U_1} = 1 | X^i_{U_2} = 1)\mathbb{P}(X^i_{U_2} = 1) \\ &= \mathbb{P}(X^i_{U_1} = 1 | X^i_{U_2} = 1)\tilde{p}_{U_2}\end{aligned}$$

In order to estimate a better bound to the quantity $\mathbb{E}[X^i_{U_1}X^i_{U_2}]$ we need to exploit the value of $\mathbb{P}(X^i_{U_1} = 1 | X^i_{U_2} = 1), U_1, U_2 \in \mathcal{U}, i = 1, \ldots, s$, in particular we observe that once we know that $X^i_{U_2} = 1$ this gives us the following information: "t_r chosen at random in the current iteration is in the interval $[t_l^{U_2} - c\delta, t_1^{U_2}]$"; this immediately restricts the possible motifs $U_1 \in \mathcal{U}$ that can satisfy $\mathbb{P}(X^i_{U_1} = 1 | X^i_{U_2} = 1)$. In particular the motifs instances of interest are the only instances $U_1 \in \mathcal{U}$ for which it holds $(t_1^{U_1} \geq t_l^{U_2} - c\delta) \wedge (t_l^{U_1} \leq t_1^{U_2} + c\delta)$, let call such set of motifs $\mathcal{U}_{U_2}$ where we refer to $\mathcal{U}_{U_a} = \{U \in \mathcal{U} : (t_1^U \geq t_l^{U_a} - c\delta) \wedge (t_l^U \leq t_1^{U_a} + c\delta) \wedge U \neq U_a\}$.

To bound the probability $\mathbb{P}(X^i_{U_1} = 1 | X^i_{U_2} = 1), U_1 \in \mathcal{U}_{U_2}$ we need to exploit a partition of the set $\mathcal{U}_{U_2}$, the partition is achieved looking at the values of $t_1^{U_1}$ and $t_l^{U_1}$ w.r.t., $t_1^{U_2}$ and $t_l^{U_2}, \forall U_1 \in \mathcal{U}_{U_2}$ in particular we distinguish the following sets:

- $\mathcal{U}^1_{U_2} = \{U_1 \in \mathcal{U}_{U_2} : (t_1^{U_1} < t_1^{U_2}) \wedge (t_l^{U_1} < t_l^{U_2})\}$;
- $\mathcal{U}^2_{U_2} = \{U_1 \in \mathcal{U}_{U_2} : (t_1^{U_1} < t_1^{U_2}) \wedge (t_l^{U_1} \geq t_l^{U_2})\}$;
- $\mathcal{U}^3_{U_2} = \{U_1 \in \mathcal{U}_{U_2} : (t_1^{U_1} \geq t_1^{U_2}) \wedge (t_l^{U_1} \leq t_l^{U_2})\}$;
- $\mathcal{U}^4_{U_2} = \{U_1 \in \mathcal{U}_{U_2} : (t_1^{U_1} \geq t_1^{U_2}) \wedge (t_l^{U_1} > t_l^{U_2})\}$.

Observe that some of these sets may be empty, but such division helps to estimate the probability $\mathbb{P}(X^i_{U_1} = 1 | X^i_{U_2} = 1), U_1 \in \mathcal{U}_{U_2}$ in particular such probability is equal to:

- $p_1^{U_1} = (c\delta - \Delta(U_1) - t_l^{U_2} + t_l^{U_1})/(c\delta - \Delta(U_2))$ if $U_1 \in \mathcal{U}^1_{U_2}$;
- $p_2^{U_1} = (c\delta - \Delta(U_1))/(c\delta - \Delta(U_2))$ if $U_1 \in \mathcal{U}^2_{U_2}$;
- 1 if $U_1 \in \mathcal{U}^3_{U_2}$;
- $p_4^{U_1} = (c\delta - \Delta(U_1) - t_1^{U_1} + t_1^{U_2})/(c\delta - \Delta(U_2))$ if $U_1 \in \mathcal{U}^4_{U_2}$;

We observe that the major part of such probabilities may be lower than one, and only in the case where $U_1 \in \mathcal{U}^3_{U_2}$ the probability is always equal to one. We now define the following quantities which we will need to bound the value of $\mathbb{E}[X_i^2], i = 1, \ldots, s$, let,

$$\hat{p}_{\mathcal{U}^j_U} = \max_{U_1 \in \mathcal{U}^j_U} \{p_j^{U_1}\}, j = 1, 2, 4$$

and let

$$\hat{C}^*_M = \max_{U \in \mathcal{U}} \{\hat{p}_{\mathcal{U}^1_U} |\mathcal{U}^1_U| + \hat{p}_{\mathcal{U}^2_U} |\mathcal{U}^2_U| + |\mathcal{U}^3_U| + \hat{p}_{\mathcal{U}^4_U} |\mathcal{U}^4_U|\}$$

then the following lemma holds,

Lemma 7. *Let $X_i, i = 1, \ldots, s$ be defined as in Algorithm 9 then it holds:*

$$\mathbb{E}[X_i^2] \leq \frac{\Delta_{\mathcal{T},2}}{(c-1)\delta} C_M \hat{C}^*_M$$

Proof. By definition of $\mathbb{E}[X_i^2]$ we have,

$$\mathbb{E}[X_i^2] = \sum_{U_1\in\mathcal{U}}\sum_{U_2\in\mathcal{U}} \frac{1}{\tilde{p}_{U_1}\tilde{p}_{U_2}}\mathbb{E}[X^i_{U_1}X^i_{U_2}] \overset{(1)}{=} \sum_{U_1\in\mathcal{U}}\sum_{U_2\in\mathcal{U}} \frac{1}{\tilde{p}_{U_1}}\mathbb{P}(X^i_{U_1}=1|X^i_{U_2}=1)$$

$$\overset{(2)}{\leq} \frac{\Delta_{\mathcal{T},2}}{(c-1)\delta}\sum_{U_2\in\mathcal{U}}\left(\sum_{U_1\in\mathcal{U}^1_{U_2}} p_1^{U_1} + \sum_{U_1\in\mathcal{U}^2_{U_2}} p_2^{U_1} + \sum_{U_1\in\mathcal{U}^3_{U_2}} 1 + \sum_{U_1\in\mathcal{U}^4_{U_2}} p_4^{U_1}\right)$$

$$\overset{(3)}{\leq} \frac{\Delta_{\mathcal{T},2}}{(c-1)\delta}\sum_{U_2\in\mathcal{U}}\left(\hat{p}_{\mathcal{U}^1_{U_2}}\sum_{U_1\in\mathcal{U}^1_{U_2}} 1 + \hat{p}_{\mathcal{U}^2_{U_2}}\sum_{U_1\in\mathcal{U}^2_{U_2}} 1 + \sum_{U_1\in\mathcal{U}^3_{U_2}} 1 + \hat{p}_{\mathcal{U}^4_{U_2}}\sum_{U_1\in\mathcal{U}^4_{U_2}} 1\right)$$

$$\overset{(4)}{=} \frac{\Delta_{\mathcal{T},2}}{(c-1)\delta}\sum_{U_2\in\mathcal{U}}\left(\hat{p}_{\mathcal{U}^1_{U_2}}|\mathcal{U}^1_{U_2}| + \hat{p}_{\mathcal{U}^2_{U_2}}|\mathcal{U}^2_{U_2}| + |\mathcal{U}^3_{U_2}| + \hat{p}_{\mathcal{U}^4_{U_2}}|\mathcal{U}^4_{U_2}|\right)$$

$$\overset{(5)}{\leq} \frac{\Delta_{\mathcal{T},2}}{(c-1)\delta} C_M \hat{C}^*_M$$

In step (1) we used the equality $\mathbb{E}[X^i_{U_1}X^i_{U_2}] = \mathbb{P}(X^i_{U_1}=1|X^i_{U_2}=1)\tilde{p}_{U_2}$ and we simplified $\tilde{p}_{U_2}$, in (2) we swapped the sums which can be done since the sums are finite, we applied the bound on $1/\tilde{p}_{U_1}$ and we applied the partition to the motifs $U_1 \in \mathcal{U}_{U_2}$ substituting $\mathbb{P}(X^i_{U_1}=1|X^i_{U_2}=1)$ with the value of such probability in each partition, as argued previously. In (3) we bounded each probability $p_j^{U_1}, j = 1, 2, 4$ with the maximum of such probability assumed by a motif in the respective interval. Step (4) just re-writes the inner sums as the cardinality of the specific sets. Step (5) uses the definition of $\hat{C}^*_M$ to derive the final bound. □

Lemma 8. *In algorithm 9 it holds that,* $\mathrm{Var}\left(\frac{1}{s}||\boldsymbol{X}||_1\right) = \mathrm{Var}\left(\frac{1}{s}\sum_{i=1}^s X_i\right) \leq \frac{C_M}{s}\left(\frac{\Delta_{\mathcal{T},2}\hat{C}^*_M}{(c-1)\delta} - C_M\right)$

Proof. Following the proof of lemma 4, we need to bound $\mathrm{Var}(X_i), i = 1, \ldots, s$,

$$\mathrm{Var}(X_i) = \mathbb{E}[X_i^2] - \mathbb{E}[X_i]^2 \leq \frac{\Delta_{\mathcal{T},2}}{(c-1)\delta} C_M \hat{C}^*_M - C_M^2 = C_M\left(\frac{\Delta_{\mathcal{T},2}\hat{C}^*_M}{(c-1)\delta} - C_M\right).$$

where we used the bound proved on $\mathbb{E}[X_i^2], i = 1, \ldots, s$ in lemma 5, now using the definition of the estimator we obtain,

$$\mathrm{Var}\left(\frac{1}{s}\sum_{i=1}^s X_i\right) \leq \frac{C_M}{s}\left(\frac{\Delta_{\mathcal{T},2}\hat{C}^*_M}{(c-1)\delta} - C_M\right).$$

□

3.5 Analysing the time complexity

The complexity of Algorithm 9 is dominated by the exact mining routine, for which we used the backtracking algorithm by [10], such routine as described in Chapter 2 is also used by Liu et al. [9] in their algorithmic framework. Thus our Algorithm 9 has a similar temporal complexity of BT+S, in particular it is bounded by $O(\sum_i^s |\mathcal{T}_i|^l + sC_M^{c\delta})$ where we recall $C_M^{c\delta}$ is the maximum number of motifs in a temporal interval of length $c\delta$; while the complexity becomes $O(\max_{i=1,\dots,s}\{|\mathcal{T}_i|^l\}+C_M^{c\delta})$ for a parallel implementation when enough threads are available.

For Algorithm 4 we have the additional step of computing the r_U for each motif in the sample which may be non negligible thus, let $|\mathcal{I}^*_{c\delta}|$ be the maximum number of edges of $\mathcal{E}$ in an interval of length $c\delta$, then the complexity of a serial implementation of 4 may be bounded by $O(\sum_i^s |\mathcal{T}_i|^l + sC_M^{c\delta}|\mathcal{I}^*_{c\delta}|)$ while a parallel implementation of the loop in line 6, leads to a overall complexity of $O(\max_{i=1,\dots,s}\{|\mathcal{T}_i|^l\} + C_M^{c\delta}|\mathcal{I}^*_{c\delta}|)$. Such algorithm has thus a worst asymptotic complexity than it's improved version.

Observe the impact of s in both the running times, in particular we know that increasing s results in a more accurate estimate. But a larger s increases also the running time, thus it is important to have a strict bound on such quantity to obtain the desired ϵ-approximation and have a small running time.

Chapter 4

Parallel Exact Approach

In this chapter we present the exact parallel approach we devised for mining motifs in temporal networks. Such algorithm will use two different key ingredients, the first one is the exact algorithm developed by Mackey et al. [10] which can enumerate all the possible δ-instances of a given temporal motif, without limits on the number of nodes or edge of such motif. The second ingredient is based on the approach of "partitioning" devised by Sun et al. [15] which exploits a partition of a given temporal graph in input. To understand the final algorithm we will present some definitions in the following section, then we explain the algorithm and how we improved it to be both scalable and efficient in practice.

4.1 Definitions

Definition 10. Given a temporal graph $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ and given $G_u = (V_u, E_u)$ the undirected static subgraph associated with $\mathcal{T}$, we say that $\mathcal{T}$ is *weakly connected* if $\forall (u, v) \in (V_u \times V_u) \setminus \{(v, v) : v \in V_u\}$ there exists a path from u to v and vice versa, i.e., for all the possible pair of nodes without counting the pair with the same node, there exists a path.

The following definitions come from [15] and they have been adapted to our framework,

Definition 11 (Adapted from definition 2 of [15])**.** Given a temporal graph $\mathcal{T} = (\mathcal{V}, \mathcal{E})$, given $e_1 = (u_1, v_1, t_1), e_2 = (u_2, v_2, t_2)$ such that $e_1, e_2 \in \mathcal{E}, e_1 \neq e_2$ and given $\delta \in \mathbb{R}^+$, we say that the edge e_1 is *δ-temporally related* to edge e_2 if they are temporally adjacent, i.e., $\{u_1, v_1\} \cap \{u_2, v_2\} \neq \emptyset$ and $|t_1 - t_2| \leq \delta$.

Definition 12 (Adapted from definition 3 of [15])**.** Given a temporal graph $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ and $\delta \in \mathbb{R}^+$, we say that $\mathcal{T}$ is a *δ-temporally connected graph* if and only if the graph is weakly connected and all the adjacent edges are δ-temporally related.

Definition 13 (Adapted from definition 6 of [15])**.** Given a temporal graph $\mathcal{T} = (\mathcal{V}, \mathcal{E})$ and $\delta \in \mathbb{R}^+$, $\mathcal{T}_i$ is a *δ-maximum connected subgraph* of $\mathcal{T}$ if and only if $\mathcal{T}_i$ is a δ-temporally connected subgraph of $\mathcal{T}$ and there is no other $\mathcal{T}_i'$, δ-temporally connected subgraph of $\mathcal{T}$ that is a supergraph of $\mathcal{T}_i$.

An example of the last definition, since it may not be so intuitive, is reported in figure 4.1, where in (4.1a) we have the temporal graph and in (4.1b) and (4.1c) there are the two δ-maximum connected subgraphs of the input graph.

4.2 The algorithms

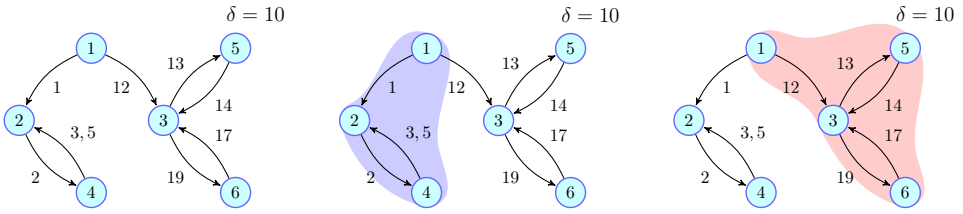
First of all we report the "partitiong algorithm" developed by [15], that is used in our parallel exact approach as a preprocessing step.

Algorithm 10: Partitioning algorithm [15]

Input: $\mathcal{T} = (\mathcal{V}, \mathcal{E}), \delta \in \mathbb{R}^+$.
Output: $\mathcal{T}_1, \ldots, \mathcal{T}_f$ where each $\mathcal{T}_i, i = 1, \ldots, f$ is a δ-maximum connected subgraph of $\mathcal{T}$.

```
1  Mark all edges of E as unprocessed
2  i ← 0
3  foreach e ∈ E do
4      if e is unprocessed then
5          Mark e as processed
6          i ← i + 1
7          T_i ← T_i ∪ {e}
8          foreach adjacent edge e_a of e do
9              if |t(e_a) − t(e)| ≤ δ then
10                 DFSPart(e_a, T_i, δ)
11 return T_1, ..., T_i
```

Where clearly the last i corresponds to f in the declaration of the output of such algorithm and the function $t(\cdot)$ returns the timestamp of the edge on which is invoked; to fully describe such approach we need to specify the routine DFSPart which is described in Algorithm 11.



(a) Temporal graph $\mathcal{T}$.

(b) First δ-maximum connected subgraph of $\mathcal{T}$.

(c) Second δ-maximum connected subgraph of $\mathcal{T}$.

Figure 4.1: Example of application of the Algorithm 10 in particular we have: (a) the input graph of the algorithm, and $\delta = 10$; (b) the first δ-maximum connected component, this component it is obtained launching the DFSPart algorithm from the edge $(1, 2, 1)$; (c) the second δ-maximum connected component, this component it is obtained launching the DFSPart algorithm from the edge $(1, 3, 12)$, that is once the DFS routine has computed the component in (b) the algorithm checks if it holds $12 - 1 \leq \delta$, since this is not verified a new component is instantiated, it is easy to see that all the other edges highlighted are in such component.

Algorithm 11: DFSPart$(e_a, \mathcal{T}_i, \delta)$ [15]

Input: e_a the edge being processed, $\mathcal{T}_i$ the current component being processed, time span $\delta \in \mathbb{R}^+$.

```
1 if e_a is not processed then
2     T_i ← T_i ∪ {e}
3     Mark e_a as processed
4     foreach adjacent edge e'_a of e_a do
5         if |t(e'_a) − t(e_a)| ≤ δ then
6             DFSPart(e'_a, T_i, δ)
```

An example of the application and the output of such algorithm is reported in figure 4.1.

The core idea of parallelizing an exact routine for counting temporal motifs is to use the already introduced Algorithm 10 to extract all the δ-maximum connected subgraphs of a given input graph, and then count on each component in parallel the number of motifs instances, summing all the partial results together. Such intuitive idea leads to Algorithm 12, where in line 1 we obtain all the δ-maximum connected subgraphs. The number f of such subgraphs may be very large, typically $f >> nt$ where nt is the number of threads available on the current machine, thus executing in parallel the exact counting on each component may lead to a very inefficient algorithm. Since many of the components are very small, the idea is to merge with a greedy procedure many of such components to obtain the new graphs

Algorithm 12: Exact Parallel Algorithm

Input: $\mathcal{T} = (\mathcal{V}, \mathcal{E}), \delta \in \mathbb{R}^+, M = (\mathcal{K}, \sigma), nt$ number of threads.
Output: C_M exact number of δ-instances of M in $\mathcal{T}$.

```
1 T_1, ..., T_f ← Partitioning Algorithm(T, δ)
2 T_1, ..., T_r ← GreedyAggregator(T_1, ..., T_f, nt)
3 for i = 1, ..., r (in parallel) do
4 |   C_M^i ← EXACT_MOTIF_COUNTER(T_i, M, δ)
5 C_M ← Σ_{i=1}^{r} C_M^i
6 return C_M
```

$\mathcal{T}_1, \ldots, \mathcal{T}_r$ where $r \sim nt$. Then we execute in parallel on each graph the exact routine (lines 3-4), as last thing we obtain the final count and return it (lines 5-6).

Now we motivate the fact that aggregating some of the δ-maximal components may lead to an improvement of performances, even though we are launching the exact routine on larger components. We use as exact routine the algorithm developed by Mackey et al. [10], such routine has an overall worst case complexity of $O(|\mathcal{E}_i|^l)$ when it is executed on the component $\mathcal{T}_i, i = 1, \ldots, r$ where we recall $l = |\mathcal{E}_\mathcal{K}|$ and $|\mathcal{E}_i|$ is the number of edges in the i-th component, $i = 1, \ldots, r$. Looking inside the Backtracking algorithm of Mackey et al. one can figure out that such algorithm is in some sense able to recognize the different components even if merged, so the time is limited by $O(\sum_j |\mathcal{E}_j|^l)$ where the sum is taken over the indexes which form the i-th component $\mathcal{T}_i, i = 1, \ldots, r$, such complexity may be much lower than $O(|\mathcal{E}_i|^l)$. Thus the overall complexity is limited by $O(\max_{\mathcal{T}_1,\ldots,\mathcal{T}_r}\{\sum_j |\mathcal{E}_j|^l\})$, where the sum is taken over the indexes which form the i-th component for $i = 1, \ldots, r$. Such complexity may be not so distant from launching all the computations in parallel if the greedy aggregator do not aggregate "large" subgraphs together. Thus, with our procedure we avoid much overhead of launching too many threads in parallel which may cause congestion and slow down significantly the algorithm.

As last thing, many approaches may be used for the greedy aggregator, ours is reported in Algorithm 13. The basic idea is to limit the number of components to $2 \cdot nt$ since with too many threads there it is a non negligible overhead for the CPU. Following such idea we define $2nt$ empty components and each step $i = 1, \ldots, f$ we put the i-th δ-maximum connected subgraph in the component which will have the minimum size after the insertion.

Algorithm 13: GreedyAggregator

Input: $\mathcal{T}_1, \ldots, \mathcal{T}_f$ δ-maximum subgraphs, nt number of threads.
Output: $\mathcal{T}_1, \ldots, \mathcal{T}_r$.

```
1 T'_1, ..., T'_2nt ← Empty components
2 for i = 1, ..., f do
3     idx ← 0; minSize ← ∞
4     for j = 1, ..., 2nt do
5         if |T'_j ∪ T_i| < minSize then
6             idx ← j
7             minSize ← |T'_j ∪ T_i|
8     T'_idx ← T'_idx ∪ T_i
9 return T'_1, ..., T'_2nt
```

Chapter 5

Experimental Evaluation

In this chapter we present the experimental evaluation we performed on several dataset coming from the SNAP library[1], such datasets are reported in table 5.1. All the experiments we are going to present were performed on a 4 core Intel 4790k CPU with 16GB of RAM.

Dataset	# of nodes	# of static edges	# of temporal edges	time span	size (MB)
CollegeMsg	1.9K	20.3K	59.8K	194 days	1,2
email-Eu-core	986	24.9K	332.3K	2,20 years	5,5
sx-SuperUser	192K	854K	1.44M	7,60 years	34,1
FBWall	45.8K	264K	856K	4,27 years	19,4
SMS-A	44.4K	-	548K	89 days	10,3
MathOverflow	24.8K	228K	390K	6,44 years	22,9
AskUbuntu	157K	545K	727K	7,16 years	10,9
Wikitalk	1.09M	3.13M	6.10M	6,23 years	173,5

Table 5.1: Temporal datasets used in the experiments.

The chapter is structured as follows,

- We discuss the results of the method BT+S by Paul Liu et al. [9] for different choice of parameters, more specifically for two values of r; with respect to the quality of the approximation;
- we discuss the results of the implementation of our two algorithm variants comparing their accuracy in the approximation also with respect to BT+S of Liu et al.;
- we compare the running times of the procedures of BT+S for the two values of r with the running times of our two sampling version;

[1] `https://snap.stanford.edu/temporal-motifs/data.html`

Shorthand	δ	c	r	b
θ'_1	86400	20	100	1
θ'_2	86400	20	30	1
θ'_3	3200	20	100	1
θ'_4	3200	20	30	1

Table 5.2: Configuration parameters used in the evaluation of the algorithm "BT+S" from Paul Liu et al.

- we discuss the quality of the approximation on the dataset Wikitalk of the parallel version of our sampling techniques and the algorithm BT+PS by Liu et al. for different values of r, with $\delta = 86400$ value for which the sequential implementations cannot handle such dataset;
- we discuss the running times of the parallel implementations of our sampling algorithms, our parallel exact routine and the algorithm BT+PS with different parameters of Liu et al.

All the tables we are going to present, report the motif $M_{i,j}, i, j = 1, \ldots, 6$ from the 6×6 grid from the article of Paranjape et al. reported in figure 2.1. We computed the exact number of δ-instances of motif $M_{i,j}$ under the column C_M. For each network, with the specific choice of parameters fixed, we executed 3 runs for each motif $M_{i,j}$, reported as run 1, 2, 3; to evaluate the random nature of the algorithms we are considering. For each of the runs we computed the error in percentage from the estimate C'_M to the true count C_M, i.e., this is obtained through $|C'_M - C_M|/C_M \cdot 100$; thus for every method and for everyone of the 3 runs such approximation is reported.

5.1 Comparison of BT+S for different values of r

Liu et al. released the code publicly[2], their methods are all implemented in C++. We focused on the implementation that use the backtracking algorithm (BT) of Mackey et al. [10] since the other methods proposed by Liu et al., i.e., EX23+S and EX23+PS, are designed only for one specific motif so they cannot be used on the whole grid we want to test. Several parameters have to be chosen, we tested the configurations reported in table 5.2. We typically chose $\delta = 86400$ which was set to 3200 only on the dataset Wikitalk since, otherwise the methods could not terminate without running out of memory. The choice of $b = 1$ is made by the Liu et al. in their source code so we do not changed such parameter, the choice of $c = 20$ it is suggested from the authors while the crucial decision is how to set r. We recall that

[2] `https://gitlab.com/paul.liu.ubc/sampling-temporal-motifs`

the probability of each interval to be sampled, i.e., q_j in the algorithm, is computed as follows $q_j = r\frac{|I_j|}{|\mathcal{E}|}$ thus it's value has a crucial impact on the number of samples accounted in their algorithm. The authors suggested to set it in $[10, 100]$, so we choose two different values 30 and 100. In the tables we are going to present we report $\bar{s}_1$ and $\bar{s}_2$ which represent, the maximum number of intervals of length $c\delta$ evaluated respectively by BT+S with $r = 100$ and $r = 30$ among the three runs. Furthermore we report φ_1 and φ_2 which reports the maximum fraction of temporal edges accounted during the three runs of the whole sampling procedure for respectively BT+S with $r = 100$ and $r = 30$, i.e., the maximum over the three runs of the following quantity $\varphi_i = \sum_j |I_j|/|\mathcal{E}|, i = 1, 2$; observe that since $b = 1$ then $\varphi_i \leq 1, i = 1, 2$ where $\varphi = 1$ means that all the edges are accounted from the algorithm.

The results are presented in the tables from page 46 to 53, we can see that in all the datasets, except Wikitalk, the version of BT+S with a higher r achieves better performances or at least not worse than the version with $r = 30$; this is not surprising in fact this is reflected in the values of $\bar{s}_1, \bar{s}_2$ and φ_1, φ_2. In the datasets where the two versions have similar quality on the approximation then the values of $\bar{s}_1, \bar{s}_2$ and φ_1, φ_2 tend to be close values, this means that approximately the same samples are used in both the procedures, while in the datasets where the version with $r = 100$ is significantly better, then such version uses much more samples than the version with $r = 30$, an example of such situation is reported in the table of the dataset SuperUser at page 48. We also observe that even if the same number of samples are accounted from the two versions, usually the version with $r = 100$ achieves better approximations since the value of r is also used to weight each motif instance, thus a higher r improves the final estimate. The version with $r = 100$ achieves good performances but in our experiments, differently from what happened in the experiments of Liu et al., sometimes the approximation error is much higher than 5%, see for example the runs for motif $M_{6,1}$ on the email-Eu-Core dataset at page 47. Finally we conclude discussing the results of the Wikitalk dataset on page 53 where we set $\delta = 3200$, where motifs $M_{4,5}$ and $M_{4,6}$ are not reported since they ran out of memory. All the two versions achieves an approximation result which is way higher than the values achieved on the other datasets, this is again not surprising since the dataset is very large and δ is small, then $|I_j|/|\mathcal{E}|$ is a small value, then multiplying it by 100 or 30 it is not sufficient to sample "enough" windows, this is reflected in the values of φ_1 and φ_2 which are very small. As we showed setting correctly the value of r has a crucial impact on the quality of approximation, and on the running times as we will see, and finding a "good" value for such parameter is not always so easy.

Approximation Factor in % on the dataset CollegeMsg											
		BT+S - θ'_1					BT+S - θ'_2				
Motif	C_M	$\bar{s}_1$	φ_1	run 1	run 2	run 3	$\bar{s}_2$	φ_2	run 1	run 2	run 3
$M_{1,1}$	487365	10	1.0	1.25%	1.54%	1.0%	10	1.0	2.81%	2.87%	2.44%
$M_{1,2}$	295970	10	1.0	1.13%	0.09%	0.09%	10	1.0	3.7%	2.44%	3.58%
$M_{1,3}$	19929	10	1.0	1.58%	1.58%	1.2%	10	1.0	2.1%	2.08%	1.83%
$M_{1,4}$	20000	10	1.0	1.52%	0.47%	1.0%	10	1.0	1.15%	1.17%	2.14%
$M_{1,5}$	861906	10	1.0	1.45%	7.41%	1.86%	10	1.0	4.56%	5.25%	3.29%
$M_{1,6}$	1204020	10	1.0	0.59%	1.58%	1.42%	10	1.0	1.17%	2.65%	2.5%
$M_{2,1}$	368884	10	1.0	12.0%	0.23%	1.0%	9	0.99	9.95%	3.65%	3.9%
$M_{2,2}$	254907	10	1.0	1.79%	0.05%	0.05%	10	1.0	4.02%	3.13%	2.05%
$M_{2,3}$	16064	10	1.0	1.13%	0.78%	1.11%	10	1.0	0.73%	0.43%	1.94%
$M_{2,4}$	9850	10	1.0	2.05%	2.05%	0.37%	10	1.0	0.92%	0.92%	0.57%
$M_{2,5}$	829831	10	1.0	1.03%	1.7%	1.57%	10	1.0	3.02%	2.2%	4.1%
$M_{2,6}$	800249	10	1.0	0.62%	3.08%	2.54%	9	0.99	1.9%	1.47%	0.96%
$M_{3,1}$	336455	10	1.0	1.49%	1.72%	1.89%	10	1.0	3.86%	3.87%	4.09%
$M_{3,2}$	349781	10	1.0	1.47%	1.47%	11.77%	9	0.99	4.21%	4.21%	9.83%
$M_{3,3}$	854505	10	1.0	5.0%	1.04%	0.17%	10	1.0	1.72%	3.7%	3.58%
$M_{3,4}$	1061197	10	1.0	5.35%	0.67%	0.43%	10	1.0	2.93%	2.52%	3.17%
$M_{3,5}$	14138	10	1.0	1.62%	2.19%	1.42%	10	1.0	2.28%	2.96%	2.38%
$M_{3,6}$	20041	10	1.0	2.7%	4.31%	1.91%	10	1.0	2.06%	3.74%	3.94%
$M_{4,1}$	711713	10	1.0	1.82%	0.85%	0.75%	10	1.0	3.52%	1.31%	2.22%
$M_{4,2}$	331604	10	1.0	0.31%	1.33%	1.12%	10	1.0	3.28%	4.43%	3.62%
$M_{4,3}$	1759008	10	1.0	2.51%	1.45%	0.79%	10	1.0	0.49%	3.56%	0.83%
$M_{4,4}$	866703	10	1.0	0.81%	0.24%	1.5%	10	1.0	2.83%	3.04%	3.94%
$M_{4,5}$	20853	10	1.0	2.16%	1.87%	2.13%	9	0.99	2.46%	2.36%	2.44%
$M_{4,6}$	17848	10	1.0	5.65%	0.97%	0.02%	10	1.0	2.72%	1.62%	0.56%
$M_{5,1}$	398228	9	0.99	2.7%	2.57%	2.57%	10	1.0	6.39%	6.15%	6.58%
$M_{5,2}$	364948	10	1.0	0.98%	0.83%	0.77%	10	1.0	6.65%	8.69%	9.41%
$M_{5,3}$	751816	10	1.0	1.66%	1.11%	1.92%	10	1.0	4.83%	4.24%	4.53%
$M_{5,4}$	891158	10	1.0	0.96%	1.19%	1.19%	10	1.0	3.34%	2.12%	2.42%
$M_{5,5}$	747568	10	1.0	0.15%	1.18%	1.75%	10	1.0	2.9%	4.26%	3.85%
$M_{5,6}$	882872	10	1.0	1.22%	1.47%	1.18%	9	0.99	0.2%	3.46%	4.09%
$M_{6,1}$	773848	10	1.0	1.36%	0.67%	1.39%	10	1.0	4.11%	4.75%	4.29%
$M_{6,2}$	381720	10	1.0	1.06%	1.06%	3.28%	9	0.99	7.09%	6.21%	6.21%
$M_{6,3}$	1697377	10	1.0	0.55%	0.71%	0.12%	10	1.0	1.91%	0.57%	1.86%
$M_{6,4}$	953679	10	1.0	0.18%	0.75%	1.66%	10	1.0	1.73%	2.53%	1.31%
$M_{6,5}$	910724	10	1.0	0.86%	0.48%	1.99%	10	1.0	2.94%	1.71%	3.25%
$M_{6,6}$	1201092	10	1.0	0.55%	0.77%	0.06%	9	0.99	1.56%	1.25%	2.59%

Approximation Factor in % on the dataset email-Eu-core											
		BT+S - θ'_1					BT+S - θ'_2				
Motif	C_M	$\bar{s}_1$	φ_1	run 1	run 2	run 3	$\bar{s}_2$	φ_2	run 1	run 2	run 3
$M_{1,1}$	514417	28	1.0	0.1%	0.34%	1.61%	24	0.92	6.93%	1.34%	3.33%
$M_{1,2}$	430620	29	1.0	0.5%	0.11%	1.45%	25	0.95	3.09%	7.09%	3.43%
$M_{1,3}$	169284	29	1.0	0.85%	0.01%	0.15%	24	0.92	3.14%	2.64%	1.9%
$M_{1,4}$	198631	28	1.0	0.46%	1.37%	0.49%	24	0.95	0.37%	0.39%	3.46%
$M_{1,5}$	626891	27	0.99	0.77%	0.99%	0.91%	24	0.95	0.78%	1.57%	0.6%
$M_{1,6}$	789842	28	1.0	0.39%	0.22%	0.04%	23	0.89	4.55%	7.11%	7.57%
$M_{2,1}$	711249	28	1.0	0.15%	0.9%	0.14%	24	0.95	0.79%	3.54%	6.73%
$M_{2,2}$	817579	29	1.0	0.66%	0.81%	1.19%	24	0.95	0.37%	4.3%	1.38%
$M_{2,3}$	160528	29	0.99	1.2%	1.12%	0.81%	23	0.92	3.61%	1.08%	1.72%
$M_{2,4}$	122157	29	1.0	1.94%	0.9%	0.33%	25	0.95	0.77%	3.25%	3.19%
$M_{2,5}$	1016020	29	1.0	0.12%	1.03%	0.16%	24	0.94	7.55%	0.82%	3.63%
$M_{2,6}$	626374	27	0.99	1.33%	1.37%	1.33%	25	0.95	6.01%	3.53%	3.49%
$M_{3,1}$	705429	29	1.0	1.23%	0.15%	0.23%	25	0.95	2.62%	3.68%	1.13%
$M_{3,2}$	466983	28	1.0	0.59%	1.07%	0.36%	25	0.96	0.62%	3.03%	0.07%
$M_{3,3}$	975941	28	1.0	0.91%	0.89%	0.08%	25	0.96	3.44%	3.37%	2.84%
$M_{3,4}$	1091657	29	1.0	1.93%	1.47%	0.58%	25	0.95	0.12%	0.03%	3.02%
$M_{3,5}$	136107	28	1.0	0.74%	0.12%	1.38%	24	0.95	3.77%	2.18%	3.89%
$M_{3,6}$	209354	28	1.0	0.4%	0.7%	0.6%	25	0.95	1.7%	3.43%	1.14%
$M_{4,1}$	3385029	28	1.0	0.54%	0.63%	0.57%	24	0.92	0.92%	2.85%	2.83%
$M_{4,2}$	858673	28	1.0	0.52%	0.3%	1.51%	25	0.95	3.4%	1.02%	4.76%
$M_{4,3}$	3693684	29	1.0	0.85%	0.9%	1.11%	25	0.95	1.1%	5.15%	0.53%
$M_{4,4}$	1094325	28	1.0	0.58%	1.13%	0.79%	24	0.93	5.74%	1.22%	6.95%
$M_{4,5}$	205742	28	1.0	0.84%	0.1%	0.44%	25	0.95	2.4%	4.62%	0.8%
$M_{4,6}$	207165	27	0.99	0.58%	1.14%	0.32%	25	0.96	2.61%	1.59%	0.14%
$M_{5,1}$	1392520	28	1.0	1.25%	0.13%	1.25%	25	0.95	2.04%	10.23%	0.42%
$M_{5,2}$	1362409	28	1.0	0.79%	0.16%	1.22%	25	0.95	2.06%	5.38%	0.27%
$M_{5,3}$	921000	28	1.0	0.28%	0.38%	0.57%	25	0.94	1.39%	0.06%	1.97%
$M_{5,4}$	631995	28	1.0	1.05%	0.53%	0.64%	25	0.95	1.01%	1.18%	3.22%
$M_{5,5}$	1072210	28	1.0	0.19%	0.3%	0.24%	25	0.95	2.77%	1.64%	2.23%
$M_{5,6}$	651653	29	1.0	0.05%	1.37%	0.05%	25	0.95	0.68%	3.42%	1.24%
$M_{6,1}$	2064940	29	0.99	9.88%	8.31%	6.4%	23	0.92	18.53%	14.35%	8.71%
$M_{6,2}$	1363113	29	0.99	4.03%	0.21%	2.99%	25	0.95	6.82%	9.93%	8.32%
$M_{6,3}$	3848912	28	1.0	1.6%	2.4%	4.27%	25	0.96	11.43%	15.93%	8.07%
$M_{6,4}$	1029608	28	1.0	2.78%	0.4%	0.82%	25	0.96	0.41%	7.28%	1.49%
$M_{6,5}$	1108292	28	1.0	2.64%	1.99%	4.91%	25	0.95	17.85%	6.5%	2.28%
$M_{6,6}$	810796	29	1.0	0.14%	1.02%	0.23%	24	0.93	2.05%	0.73%	3.86%

Approximation Factor in % on the dataset sx-SuperUser											
		BT+S - θ'_1					BT+S - θ'_2				
Motif	C_M	$\bar{s}_1$	φ_1	run 1	run 2	run 3	$\bar{s}_2$	φ_2	run 1	run 2	run 3
$M_{1,1}$	166857	103	0.85	0.04%	1.95%	0.73%	29	0.24	9.08%	9.19%	2.23%
$M_{1,2}$	89735	102	0.85	1.84%	1.91%	2.1%	29	0.24	7.91%	7.65%	6.23%
$M_{1,3}$	31181	99	0.82	2.32%	1.56%	1.5%	29	0.24	13.86%	3.85%	15.8%
$M_{1,4}$	33002	99	0.82	1.46%	1.27%	1.76%	29	0.24	11.48%	8.63%	6.54%
$M_{1,5}$	202005	101	0.84	0.32%	1.62%	1.56%	29	0.24	6.16%	6.15%	8.55%
$M_{1,6}$	388398	99	0.83	1.68%	1.81%	1.17%	29	0.24	10.49%	10.74%	9.59%
$M_{2,1}$	132965	102	0.84	0.19%	2.18%	0.09%	29	0.24	18.12%	13.86%	5.58%
$M_{2,2}$	56823	101	0.84	2.06%	0.56%	1.6%	29	0.24	9.69%	8.39%	14.76%
$M_{2,3}$	21089	103	0.85	0.4%	0.17%	2.64%	29	0.24	4.82%	16.87%	20.03%
$M_{2,4}$	6334	103	0.85	3.73%	0.56%	1.06%	29	0.24	25.05%	20.31%	15.73%
$M_{2,5}$	326259	100	0.83	1.39%	1.68%	1.87%	29	0.24	8.42%	7.93%	8.35%
$M_{2,6}$	279654	99	0.82	0.35%	0.41%	0.77%	29	0.24	7.43%	0.42%	0.41%
$M_{3,1}$	79626	99	0.82	2.14%	1.14%	0.33%	29	0.24	7.57%	1.18%	6.77%
$M_{3,2}$	129020	100	0.83	1.28%	1.13%	0.67%	29	0.24	4.45%	3.23%	3.15%
$M_{3,3}$	147975	99	0.83	1.4%	1.52%	1.29%	29	0.24	4.97%	3.86%	7.59%
$M_{3,4}$	557702	102	0.85	1.07%	2.23%	1.77%	29	0.24	12.0%	10.95%	11.33%
$M_{3,5}$	12263	101	0.84	1.6%	0.17%	1.67%	29	0.24	7.39%	10.65%	9.58%
$M_{3,6}$	24870	100	0.83	1.07%	3.04%	1.91%	29	0.24	9.44%	6.98%	7.12%
$M_{4,1}$	233400	99	0.82	0.41%	1.08%	0.13%	29	0.24	8.44%	6.1%	6.23%
$M_{4,2}$	209077	102	0.85	1.8%	0.28%	1.37%	29	0.24	14.34%	13.26%	0.62%
$M_{4,3}$	1091788	102	0.84	0.34%	1.56%	4.59%	29	0.24	5.63%	6.89%	5.94%
$M_{4,4}$	601707	99	0.82	1.86%	1.37%	2.08%	29	0.24	6.93%	8.08%	5.3%
$M_{4,5}$	22457	100	0.83	1.23%	0.17%	1.95%	29	0.24	11.52%	4.78%	8.12%
$M_{4,6}$	11976	102	0.84	1.01%	1.2%	0.06%	29	0.24	7.59%	17.25%	18.04%
$M_{5,1}$	17898	101	0.84	1.13%	1.79%	1.46%	29	0.24	14.77%	9.33%	9.97%
$M_{5,2}$	112602	99	0.82	0.12%	1.19%	1.69%	29	0.24	6.76%	5.15%	3.04%
$M_{5,3}$	549672	101	0.84	1.49%	1.26%	0.66%	29	0.24	6.51%	6.54%	6.77%
$M_{5,4}$	343289	99	0.83	1.3%	0.59%	2.35%	29	0.24	7.37%	1.8%	8.37%
$M_{5,5}$	170469	101	0.84	1.99%	0.86%	1.89%	29	0.24	4.51%	9.97%	4.19%
$M_{5,6}$	184637	99	0.82	1.73%	0.98%	2.93%	29	0.24	4.91%	3.88%	5.06%
$M_{6,1}$	167850	99	0.82	0.52%	0.04%	0.34%	29	0.24	6.95%	6.83%	6.71%
$M_{6,2}$	40629	100	0.83	1.08%	0.47%	2.53%	29	0.24	9.91%	3.48%	9.54%
$M_{6,3}$	1059493	98	0.82	1.63%	2.41%	0.64%	29	0.24	6.38%	7.46%	4.76%
$M_{6,4}$	322650	98	0.82	2.14%	1.8%	1.95%	29	0.24	7.88%	11.69%	10.31%
$M_{6,5}$	472529	99	0.82	1.77%	1.66%	2.55%	29	0.24	5.19%	9.49%	10.25%
$M_{6,6}$	396247	102	0.85	0.12%	0.69%	1.34%	29	0.24	3.47%	10.57%	9.55%

Approximation Factor in % on the dataset FBWall											
		BT+S - θ'_1					BT+S - θ'_2				
Motif	C_M	$\bar{s}_1$	φ_1	run 1	run 2	run 3	$\bar{s}_2$	φ_2	run 1	run 2	run 3
$M_{1,1}$	115233	46	0.97	0.19%	0.14%	0.16%	28	0.67	1.2%	1.46%	0.56%
$M_{1,2}$	135152	46	0.98	2.91%	2.58%	0.05%	28	0.67	4.32%	2.88%	1.47%
$M_{1,3}$	11221	47	0.98	1.09%	2.43%	0.13%	28	0.67	3.99%	2.36%	3.43%
$M_{1,4}$	12256	47	0.98	1.03%	0.89%	0.8%	27	0.68	3.96%	3.07%	2.32%
$M_{1,5}$	251005	46	0.98	0.1%	0.4%	0.38%	28	0.67	3.02%	10.35%	2.5%
$M_{1,6}$	207218	46	0.98	0.23%	0.51%	0.42%	29	0.7	2.4%	5.49%	0.7%
$M_{2,1}$	90055	47	0.98	0.75%	1.06%	0.39%	28	0.7	12.54%	12.52%	2.12%
$M_{2,2}$	117750	46	0.98	0.43%	0.51%	0.03%	28	0.68	5.5%	6.32%	5.49%
$M_{2,3}$	14439	46	0.98	0.01%	0.43%	0.43%	28	0.67	13.18%	4.45%	4.84%
$M_{2,4}$	11334	47	0.98	2.54%	0.07%	0.62%	28	0.69	1.45%	9.26%	0.33%
$M_{2,5}$	209126	46	0.98	0.08%	0.64%	0.48%	28	0.69	3.3%	2.22%	4.74%
$M_{2,6}$	273644	46	0.98	0.33%	0.57%	0.22%	29	0.7	3.27%	2.2%	4.82%
$M_{3,1}$	106457	46	0.98	0.17%	0.39%	0.5%	28	0.69	1.51%	0.37%	8.65%
$M_{3,2}$	100138	46	0.98	0.59%	0.88%	0.33%	28	0.69	3.65%	3.9%	4.2%
$M_{3,3}$	212592	47	0.98	0.62%	0.05%	0.36%	28	0.69	3.27%	2.47%	5.35%
$M_{3,4}$	155864	46	0.98	0.19%	0.91%	0.17%	28	0.68	1.53%	4.55%	4.02%
$M_{3,5}$	7954	47	0.98	1.1%	0.63%	1.37%	28	0.69	11.06%	11.78%	4.58%
$M_{3,6}$	10501	46	0.97	0.47%	0.65%	1.18%	28	0.68	0.34%	1.82%	0.4%
$M_{4,1}$	154215	46	0.97	1.0%	0.89%	0.87%	28	0.68	1.86%	1.32%	2.15%
$M_{4,2}$	102091	46	0.98	1.03%	0.26%	0.26%	28	0.7	6.25%	2.33%	7.01%
$M_{4,3}$	265096	46	0.97	1.18%	0.12%	0.06%	28	0.68	0.72%	0.12%	0.36%
$M_{4,4}$	267087	46	0.98	0.19%	0.85%	0.43%	28	0.68	2.31%	2.92%	5.6%
$M_{4,5}$	13872	46	0.98	0.83%	0.82%	0.08%	28	0.67	4.15%	12.58%	10.64%
$M_{4,6}$	13074	46	0.98	1.62%	0.76%	0.88%	28	0.67	4.82%	1.23%	6.77%
$M_{5,1}$	970486	47	0.98	0.72%	0.2%	0.77%	28	0.67	3.67%	3.23%	3.33%
$M_{5,2}$	832672	47	0.98	0.89%	0.86%	0.2%	28	0.7	2.43%	6.25%	7.19%
$M_{5,3}$	293826	47	0.98	0.33%	1.13%	0.6%	28	0.67	3.22%	4.48%	5.15%
$M_{5,4}$	273445	47	0.98	0.74%	0.21%	0.61%	27	0.68	7.08%	6.75%	7.39%
$M_{5,5}$	243042	46	0.98	0.58%	0.6%	0.63%	28	0.7	3.83%	4.59%	5.19%
$M_{5,6}$	215511	47	0.98	0.11%	0.23%	0.46%	29	0.7	2.79%	4.75%	3.81%
$M_{6,1}$	939754	47	0.98	0.99%	0.07%	1.05%	28	0.67	7.25%	0.51%	6.29%
$M_{6,2}$	841299	46	0.97	0.32%	0.79%	0.29%	28	0.67	2.8%	3.76%	1.88%
$M_{6,3}$	270734	46	0.98	9.94%	1.07%	0.36%	28	0.68	5.9%	0.36%	3.45%
$M_{6,4}$	191214	47	0.98	0.18%	2.23%	0.59%	28	0.69	3.27%	1.83%	4.7%
$M_{6,5}$	185951	47	0.98	0.67%	0.55%	1.32%	28	0.69	0.94%	6.4%	6.89%
$M_{6,6}$	200650	46	0.97	0.35%	0.18%	0.08%	28	0.67	5.61%	5.8%	4.88%

Approximation Factor in % on the dataset SMS-ME											
		BT+S - θ'_1					BT+S - θ'_2				
Motif	C_M	$\bar{s}_1$	φ_1	run 1	run 2	run 3	$\bar{s}_2$	φ_2	run 1	run 2	run 3
$M_{1,1}$	1011752	19	1.0	2.16%	1.94%	1.64%	19	1.0	2.17%	2.06%	0.91%
$M_{1,2}$	1129270	18	1.0	0.82%	0.7%	1.47%	19	1.0	1.01%	1.15%	1.47%
$M_{1,3}$	21587	19	1.0	1.64%	2.0%	2.11%	19	1.0	1.64%	0.85%	6.61%
$M_{1,4}$	22676	19	1.0	0.51%	1.83%	1.53%	18	1.0	1.44%	1.87%	2.23%
$M_{1,5}$	5315944	19	1.0	1.86%	2.07%	2.07%	18	1.0	2.08%	1.52%	2.14%
$M_{1,6}$	11469761	18	1.0	0.41%	2.66%	2.66%	18	1.0	2.49%	3.24%	2.51%
$M_{2,1}$	1492053	18	1.0	0.86%	0.53%	1.39%	19	1.0	1.41%	0.89%	1.59%
$M_{2,2}$	1705749	19	1.0	1.93%	7.61%	1.51%	18	1.0	0.09%	2.4%	1.98%
$M_{2,3}$	20745	19	1.0	2.49%	2.64%	0.19%	18	1.0	0.24%	2.02%	2.71%
$M_{2,4}$	18587	19	1.0	1.81%	1.77%	4.51%	18	1.0	2.79%	0.9%	0.45%
$M_{2,5}$	5945908	19	1.0	2.02%	2.14%	1.9%	18	1.0	4.19%	8.28%	2.2%
$M_{2,6}$	4862269	19	1.0	1.16%	25.34%	1.77%	19	1.0	1.94%	1.76%	1.82%
$M_{3,1}$	1404994	19	1.0	1.58%	1.18%	0.4%	19	1.0	1.68%	0.67%	0.73%
$M_{3,2}$	1074065	18	1.0	1.27%	1.25%	0.31%	19	1.0	1.27%	1.28%	0.65%
$M_{3,3}$	6053318	19	1.0	2.07%	2.11%	2.17%	18	1.0	1.72%	1.72%	2.05%
$M_{3,4}$	5342704	19	1.0	1.6%	1.91%	2.18%	18	1.0	0.38%	1.35%	1.65%
$M_{3,5}$	20847	19	1.0	2.85%	4.61%	2.64%	19	1.0	1.82%	0.05%	2.53%
$M_{3,6}$	24213	19	1.0	0.53%	0.48%	1.39%	19	1.0	0.83%	1.27%	1.17%
$M_{4,1}$	2562388	19	1.0	0.53%	0.88%	0.33%	19	1.0	1.5%	1.8%	1.57%
$M_{4,2}$	1656174	18	1.0	0.23%	0.7%	0.81%	19	1.0	0.44%	0.9%	1.65%
$M_{4,3}$	8983423	19	1.0	0.13%	2.06%	1.93%	19	1.0	1.93%	0.44%	1.66%
$M_{4,4}$	5739729	18	1.0	1.41%	1.48%	5.11%	18	1.0	5.3%	1.97%	2.0%
$M_{4,5}$	23769	18	1.0	1.09%	0.47%	1.3%	18	1.0	1.55%	1.0%	1.47%
$M_{4,6}$	23612	19	1.0	1.17%	1.58%	2.38%	19	1.0	0.01%	0.0%	3.76%
$M_{5,1}$	54657431	19	1.0	0.54%	0.33%	1.35%	19	1.0	1.92%	1.05%	0.95%
$M_{5,2}$	54016279	18	1.0	1.52%	0.6%	1.35%	18	1.0	1.64%	2.08%	1.39%
$M_{5,3}$	5527281	19	1.0	1.65%	0.43%	2.01%	18	1.0	0.89%	1.92%	1.97%
$M_{5,4}$	3891704	19	1.0	1.2%	37.62%	1.21%	19	1.0	1.8%	1.92%	1.79%
$M_{5,5}$	5113195	19	1.0	2.1%	1.41%	1.78%	19	1.0	1.97%	1.96%	1.31%
$M_{5,6}$	3679849	19	1.0	1.21%	1.84%	4.18%	19	1.0	0.94%	1.99%	0.13%
$M_{6,1}$	84304364	19	1.0	1.25%	1.27%	0.38%	18	1.0	0.17%	1.6%	1.25%
$M_{6,2}$	53917340	19	1.0	9.24%	0.37%	1.03%	19	1.0	1.62%	1.59%	0.61%
$M_{6,3}$	8277785	18	1.0	2.01%	2.11%	0.96%	18	1.0	2.45%	2.46%	1.49%
$M_{6,4}$	4260867	19	1.0	1.76%	1.62%	1.98%	18	1.0	2.14%	1.91%	1.71%
$M_{6,5}$	4910880	19	1.0	0.48%	1.86%	2.18%	18	1.0	1.93%	1.86%	1.68%
$M_{6,6}$	4858353	19	1.0	1.55%	12.26%	1.91%	19	1.0	1.66%	2.01%	2.12%

Approximation Factor in % on the dataset MathOverflow											
		BT+S - θ'_1					BT+S - θ'_2				
Motif	C_M	$\bar{s}_1$	φ_1	run 1	run 2	run 3	$\bar{s}_2$	φ_2	run 1	run 2	run 3
$M_{1,1}$	131573	94	0.83	3.65%	0.78%	2.34%	29	0.26	2.22%	7.05%	12.11%
$M_{1,2}$	44890	94	0.83	0.76%	2.53%	2.94%	28	0.26	9.58%	7.4%	3.55%
$M_{1,3}$	26091	94	0.83	1.43%	0.0%	1.16%	29	0.26	8.34%	4.08%	10.61%
$M_{1,4}$	21087	92	0.82	1.1%	2.58%	1.02%	29	0.26	5.4%	4.91%	1.23%
$M_{1,5}$	99043	93	0.82	1.41%	1.94%	0.91%	30	0.28	12.15%	0.86%	1.09%
$M_{1,6}$	217732	94	0.83	1.89%	3.08%	0.24%	28	0.26	7.42%	0.85%	8.4%
$M_{2,1}$	34272	93	0.83	2.03%	0.63%	1.4%	29	0.26	0.23%	3.87%	4.92%
$M_{2,2}$	22100	92	0.82	2.68%	0.88%	2.58%	29	0.26	5.56%	14.59%	5.57%
$M_{2,3}$	13368	94	0.83	3.59%	0.44%	2.09%	29	0.26	3.45%	4.77%	5.84%
$M_{2,4}$	3653	94	0.83	0.25%	11.24%	0.34%	29	0.26	14.66%	8.64%	15.49%
$M_{2,5}$	101393	94	0.83	1.73%	0.7%	1.42%	29	0.26	8.82%	10.54%	0.75%
$M_{2,6}$	40368	92	0.82	0.11%	1.89%	1.1%	29	0.26	15.11%	13.09%	5.61%
$M_{3,1}$	34576	90	0.8	0.47%	2.98%	2.8%	29	0.26	1.5%	4.6%	9.33%
$M_{3,2}$	41057	93	0.82	1.32%	1.57%	1.52%	29	0.26	9.37%	0.51%	4.04%
$M_{3,3}$	52813	94	0.83	1.42%	8.38%	2.06%	29	0.26	0.35%	10.82%	3.85%
$M_{3,4}$	116977	94	0.83	1.75%	0.06%	1.31%	29	0.26	1.21%	1.94%	0.34%
$M_{3,5}$	8247	93	0.82	0.56%	0.86%	0.77%	29	0.26	12.03%	5.96%	3.52%
$M_{3,6}$	16182	94	0.83	0.8%	0.93%	2.34%	29	0.26	0.3%	8.79%	0.91%
$M_{4,1}$	54579	93	0.83	0.96%	2.28%	0.84%	29	0.26	1.41%	11.9%	1.28%
$M_{4,2}$	29390	94	0.83	5.13%	0.48%	1.19%	29	0.26	7.93%	0.68%	0.72%
$M_{4,3}$	163441	93	0.83	2.41%	1.44%	2.37%	30	0.28	7.68%	9.56%	3.2%
$M_{4,4}$	56309	94	0.83	0.97%	3.61%	4.37%	29	0.26	4.78%	4.07%	2.94%
$M_{4,5}$	14552	92	0.82	0.8%	1.89%	1.16%	29	0.26	6.75%	1.1%	1.31%
$M_{4,6}$	7518	93	0.83	7.54%	2.52%	0.11%	28	0.25	2.3%	2.49%	11.38%
$M_{5,1}$	9184	93	0.83	2.63%	1.08%	0.9%	29	0.26	1.8%	4.57%	8.36%
$M_{5,2}$	32226	94	0.83	0.06%	5.71%	3.49%	29	0.26	13.44%	12.44%	5.94%
$M_{5,3}$	54195	94	0.83	3.86%	1.02%	3.47%	29	0.26	8.2%	3.47%	2.86%
$M_{5,4}$	62683	94	0.83	1.96%	2.21%	1.2%	29	0.26	10.6%	2.68%	0.07%
$M_{5,5}$	56811	90	0.8	1.05%	0.7%	0.75%	30	0.28	22.76%	18.86%	1.31%
$M_{5,6}$	78042	93	0.83	0.44%	1.06%	0.67%	29	0.26	2.85%	17.87%	3.96%
$M_{6,1}$	91919	94	0.83	1.92%	2.37%	0.19%	29	0.26	26.63%	25.88%	27.92%
$M_{6,2}$	18407	93	0.83	5.72%	4.46%	0.31%	29	0.26	0.48%	0.97%	8.44%
$M_{6,3}$	163078	92	0.82	1.49%	1.95%	1.66%	29	0.26	7.2%	11.2%	10.56%
$M_{6,4}$	108044	91	0.82	1.99%	3.04%	3.41%	29	0.26	4.34%	4.8%	3.55%
$M_{6,5}$	96115	94	0.83	1.71%	1.64%	1.6%	29	0.26	2.25%	5.07%	5.76%
$M_{6,6}$	217864	94	0.83	0.77%	3.82%	0.48%	29	0.26	9.03%	6.3%	10.87%

Approximation Factor in % on the dataset AskUbuntu											
		BT+S - θ'_1					BT+S - θ'_2				
Motif	C_M	$\bar{s}_1$	φ_1	run 1	run 2	run 3	$\bar{s}_2$	φ_2	run 1	run 2	run 3
$M_{1,1}$	126939	89	0.93	1.33%	0.85%	3.74%	26	0.25	19.01%	21.79%	30.05%
$M_{1,2}$	63189	88	0.92	2.26%	2.56%	2.96%	27	0.26	3.64%	21.32%	4.9%
$M_{1,3}$	19951	89	0.93	8.54%	3.23%	1.41%	27	0.27	9.49%	1.97%	8.12%
$M_{1,4}$	19804	89	0.93	4.15%	5.11%	2.48%	27	0.26	1.31%	11.33%	12.3%
$M_{1,5}$	168096	88	0.92	1.56%	0.15%	2.95%	27	0.26	12.43%	13.75%	20.01%
$M_{1,6}$	426148	89	0.93	0.54%	0.48%	1.71%	27	0.26	27.87%	25.57%	28.38%
$M_{2,1}$	139655	88	0.92	0.38%	0.89%	0.28%	27	0.27	27.7%	33.59%	28.16%
$M_{2,2}$	55458	89	0.93	2.24%	2.24%	1.97%	27	0.27	37.14%	13.91%	35.89%
$M_{2,3}$	13713	88	0.93	1.46%	3.67%	5.62%	27	0.26	18.07%	12.93%	1.87%
$M_{2,4}$	5509	88	0.92	8.22%	6.68%	3.36%	28	0.27	19.17%	32.11%	15.0%
$M_{2,5}$	413142	89	0.92	0.06%	1.78%	2.81%	27	0.27	30.38%	32.13%	32.11%
$M_{2,6}$	298509	89	0.92	1.25%	0.82%	0.57%	27	0.26	24.78%	27.55%	30.58%
$M_{3,1}$	79470	88	0.92	1.29%	1.15%	2.62%	28	0.27	25.39%	22.49%	21.38%
$M_{3,2}$	113270	89	0.93	0.83%	0.58%	0.93%	27	0.26	22.35%	20.67%	14.27%
$M_{3,3}$	179073	89	0.93	2.44%	1.88%	0.67%	27	0.26	28.59%	18.17%	20.96%
$M_{3,4}$	745480	90	0.94	0.12%	0.34%	0.86%	27	0.27	23.31%	30.1%	21.6%
$M_{3,5}$	8946	89	0.93	5.26%	4.23%	2.22%	28	0.27	8.02%	1.59%	7.82%
$M_{3,6}$	14639	89	0.93	0.45%	2.66%	5.65%	27	0.26	4.17%	6.06%	5.08%
$M_{4,1}$	199595	89	0.93	1.71%	0.44%	0.01%	27	0.27	32.54%	25.49%	32.11%
$M_{4,2}$	207846	89	0.93	0.09%	0.55%	0.59%	27	0.26	22.24%	26.83%	23.5%
$M_{4,3}$	1044513	89	0.93	0.98%	0.37%	0.29%	28	0.27	44.8%	37.38%	40.63%
$M_{4,4}$	660015	89	0.92	0.61%	0.38%	0.35%	27	0.26	27.51%	31.73%	33.03%
$M_{4,5}$	15512	88	0.93	3.08%	0.16%	5.03%	27	0.26	12.13%	11.94%	5.8%
$M_{4,6}$	8346	89	0.93	7.08%	6.17%	7.33%	27	0.27	0.92%	11.93%	1.35%
$M_{5,1}$	21737	88	0.93	1.15%	2.24%	1.82%	27	0.26	28.39%	46.76%	18.38%
$M_{5,2}$	133038	89	0.93	0.54%	0.38%	0.4%	27	0.27	20.64%	22.4%	29.45%
$M_{5,3}$	613592	89	0.93	0.0%	1.97%	0.83%	27	0.27	23.88%	29.12%	23.9%
$M_{5,4}$	344810	89	0.93	0.47%	1.48%	1.36%	27	0.26	22.53%	21.25%	26.27%
$M_{5,5}$	184475	88	0.93	0.23%	1.71%	0.99%	27	0.27	29.66%	27.98%	16.17%
$M_{5,6}$	152517	88	0.93	2.72%	2.33%	0.72%	28	0.27	7.9%	3.08%	6.36%
$M_{6,1}$	248091	89	0.93	0.1%	1.02%	0.63%	27	0.26	30.52%	31.13%	31.21%
$M_{6,2}$	54097	88	0.92	1.27%	1.27%	0.76%	27	0.26	38.02%	37.18%	24.12%
$M_{6,3}$	994099	88	0.92	0.36%	2.53%	0.56%	27	0.27	43.52%	34.73%	39.01%
$M_{6,4}$	410366	90	0.94	0.72%	0.03%	1.92%	27	0.27	25.19%	25.75%	26.39%
$M_{6,5}$	654357	88	0.92	0.44%	0.44%	0.36%	27	0.26	31.41%	26.65%	29.5%
$M_{6,6}$	438895	88	0.93	1.2%	1.41%	1.77%	27	0.26	18.84%	26.26%	27.26%

Approximation Factor in % on the dataset WikiTalk

		BT+S - θ'_3					BT+S - θ'_4				
Motif	C_M	$\bar{s}_1$	φ_1	run 1	run 2	run 3	$\bar{s}_2$	φ_2	run 1	run 2	run 3
$M_{1,1}$	539857	103	0.09	13.64%	0.96%	4.53%	22	0.02	44.54%	41.81%	51.01%
$M_{1,2}$	135226	102	0.1	17.09%	16.31%	4.73%	23	0.02	20.6%	23.63%	21.77%
$M_{1,3}$	86836	103	0.09	0.27%	21.9%	6.85%	22	0.02	41.1%	42.27%	18.14%
$M_{1,4}$	100824	103	0.09	11.45%	14.86%	14.01%	23	0.02	4.25%	29.06%	36.69%
$M_{1,5}$	266910	102	0.09	9.3%	8.48%	6.44%	24	0.02	19.22%	13.22%	35.22%
$M_{1,6}$	932924	100	0.09	4.45%	21.91%	15.74%	23	0.02	32.93%	41.81%	30.07%
$M_{2,1}$	446410	99	0.09	31.72%	14.43%	8.32%	23	0.02	24.68%	173.3%	45.56%
$M_{2,2}$	248332	102	0.1	68.03%	76.57%	12.73%	26	0.03	33.66%	39.14%	19.71%
$M_{2,3}$	56354	102	0.1	12.15%	13.78%	13.39%	23	0.02	24.61%	9.92%	9.99%
$M_{2,4}$	21941	103	0.09	31.12%	20.31%	8.26%	23	0.02	17.95%	61.49%	14.41%
$M_{2,5}$	1017969	100	0.09	20.82%	23.62%	0.0%	25	0.03	29.33%	18.08%	29.29%
$M_{2,6}$	229169	99	0.09	1.08%	12.22%	1.95%	21	0.02	8.52%	13.27%	37.39%
$M_{3,1}$	462985	103	0.09	44.7%	16.96%	9.47%	23	0.02	29.23%	158.48%	159.63%
$M_{3,2}$	158717	104	0.09	6.08%	6.49%	2.08%	24	0.02	39.26%	32.53%	34.46%
$M_{3,3}$	831400	103	0.1	5.07%	6.87%	29.77%	26	0.03	15.51%	23.55%	28.38%
$M_{3,4}$	1437961	101	0.09	36.11%	14.01%	34.55%	23	0.02	55.72%	51.69%	5.22%
$M_{3,5}$	31988	103	0.09	7.99%	7.89%	8.04%	23	0.02	2.96%	52.33%	12.13%
$M_{3,6}$	78717	101	0.09	0.24%	10.17%	12.31%	22	0.02	11.74%	19.77%	10.15%
$M_{4,1}$	176665319	101	0.09	7.41%	34.99%	57.22%	24	0.02	81.3%	86.0%	42.31%
$M_{4,2}$	579299	102	0.1	14.53%	9.43%	6.45%	24	0.02	38.96%	6.61%	39.0%
$M_{4,3}$	376601375	102	0.09	18.52%	21.59%	12.07%	23	0.02	19.37%	75.37%	39.14%
$M_{4,4}$	991003	101	0.09	22.6%	3.19%	3.11%	23	0.02	8.2%	14.46%	25.14%
$M_{4,5}$	-	-	-	-	-	-	-	-	-	-	-
$M_{4,6}$	-	-	-	-	-	-	-	-	-	-	-
$M_{5,1}$	918754	103	0.09	5.21%	5.23%	8.58%	23	0.02	35.97%	28.68%	34.0%
$M_{5,2}$	825696	102	0.09	5.42%	2.35%	2.32%	26	0.03	22.02%	30.25%	34.95%
$M_{5,3}$	713196	100	0.09	9.93%	30.9%	2.22%	23	0.02	16.65%	16.62%	25.24%
$M_{5,4}$	305617	104	0.09	5.96%	8.21%	6.18%	22	0.02	16.72%	35.07%	31.61%
$M_{5,5}$	655935	101	0.09	2.55%	6.39%	1.63%	23	0.02	24.4%	20.15%	9.43%
$M_{5,6}$	314878	101	0.09	2.4%	8.49%	3.16%	26	0.03	24.89%	43.51%	18.9%
$M_{6,1}$	15047345	100	0.09	22.02%	5.87%	13.92%	23	0.02	44.69%	32.98%	70.22%
$M_{6,2}$	799080	102	0.1	7.62%	7.19%	45.59%	23	0.02	32.4%	34.86%	27.3%
$M_{6,3}$	382034965	101	0.09	16.25%	18.35%	12.93%	23	0.02	76.65%	67.49%	275.9%
$M_{6,4}$	1538217	101	0.09	0.93%	49.76%	99.21%	21	0.02	45.06%	25.96%	72.65%
$M_{6,5}$	910740	101	0.09	38.54%	35.17%	43.85%	23	0.02	24.53%	49.85%	53.14%
$M_{6,6}$	972272	102	0.1	19.32%	71.58%	1.64%	22	0.02	48.73%	43.33%	43.87%

5.2 Evaluation of our Sampling Algorithms

We implemented our two sampling versions using the C++ language, we used as base code the code of Liu et al., and tested them on the dataset in table 5.1. Our sampling algorithms are (ϵ, η)-approximation algorithms and we tested the theoretical sample sizes to obtain an $(\epsilon = 0.1, \eta = 0.05)$-approximation but the theoretical sample sizes result in a huge number of samples s which we decided to not test since it would require much more running time than an exact routine. In the experiments we performed we set the number of samples s the same for the two procedures and given a dataset and a motif $M_{i,j}, i, j = 1, \ldots, 6$ we set s to the value $\bar{s}_1$ on the same dataset and the same motif from the previous tables, such that we also may compare our algorithms to the sampling schema of Liu et al. We set the other parameters to $c = 20, \delta = 86400$ for all the datasets except wikitalk were we set $\delta = 3200$. The value of s is also reported in each table. In the tables we also reported $\bar{\phi}_i, i = 1, 2$ which are computed as follows $\bar{\phi}_i = 1/3 \sum_{j=1}^{3} \phi_j^i, i = 1, 2$, where $\phi_j^i = \sum_{e \in \mathcal{E}} \mathbb{1}[e \in \mathcal{S}_{ij}]/|\mathcal{E}|, i = 1, 2, j = 1, 2, 3$ where $\mathbb{1}[\cdot]$ is the indicator function, $\mathcal{S}_{ij} = \{e \in \mathcal{E} : e \in \mathcal{T}_a^{ij}, a = 1, \ldots, s\}$, and $\mathcal{T}_a^{ij}$ is the sample $a = 1, \ldots, s$ of variant $i = 1, 2$ at run $j = 1, 2, 3$. Intuitively the value of $0 \leq \phi_j^i \leq 1, i = 1, 2, j = 1, 2, 3$ quantifies how much the whole graph is explored by our sampling procedures, i.e., when we chose randomly an interval of length $c\delta$ we may end up choosing very close intervals, then in such case $\phi_j^i \sim 0$, if instead we covered all the possible temporal edges then $\phi_j^i \sim 1$. Then $\bar{\phi}_i, i = 1, 2$ is the mean over the three runs for each variant of the quantities ϕ_j^i. Thus informally, such number represents the fraction of spanned edges, i.e., an edge is spanned if it is contained in some sample, during the sampling procedure.

In the tables from page 55 to 62 we present our results, in particular we first observe that it is not so clear which of the two procedures performs better since this also depends on the motif and on the dataset. We observe that our procedures achieve very high approximations on the datasets CollegeMsg, email-Eu-core and SMS-ME and for some motifs on FBWall. While the approximation is not so high on the other datasets except Wikitalk which we will discuss at the end. Comparing our procedures to the ones of Liu et al. we observe that their procedures work much better, this is also due to great variance our estimates have (see the variance analysis in Chapter 3), while the estimate of their schema has lower variance (see Chapter 2). The great variance of our algorithms is reflected in the data, one example is the second variant for motif $M_{6,2}$ in the dataset email-Eu-Core at page 56 which ranges from an approximation factor of $0,82\%$ to $74,37\%$ in only three runs. Finally, looking at the wikitalk dataset where motifs $M_{4,5}$ and $M_{4,6}$ are not reported since they ran out of memory, our algorithms are, except for some "unlucky" motifs, comparable to the algorithms of Liu et al., which is interesting since

we have greater variance but also achieve some lower approximation factor than their procedures.

Looking at the values of $\bar{\phi}_i, i = 1, 2$ we observe that usually our procedures have such value under 0.8, this means that there are some intervals which we do not look for in the whole procedure, this may be the key to understand the large variance of our estimate. In particular we may end up sampling many intervals which do not contain the motifs we are looking for, and we may not look at the only important intervals we need. Moreover, we observe that a higher value of $\bar{\phi}_i, i = 1, 2$ leads to a lower mean approximation factor along the three runs; which reconciles with the fact that more we explore the dataset, a better estimate we may get. Interestingly on the dataset SuperUser, table at page 58, we achieve good approximation factors overall but the values of $\bar{\phi}_i, i = 1, 2$ ae under 0.6 which suggests that this value alone cannot explain all the performances of our algorithms, thus further measures may be needed.

Approximation Factor in % on the dataset CollegeMsg										
			First Variant				Second Variant			
Motif	C_M	s	$\bar{\phi}_1$	run 1	run 2	run 3	$\bar{\phi}_2$	run 1	run 2	run 3
$M_{1,1}$	487365	10	0.77	66.77%	28.62%	76.63%	0.8	5.94%	139.65%	47.34%
$M_{1,2}$	295970	10	0.78	3.28%	32.38%	7.84%	0.88	17.71%	17.53%	86.97%
$M_{1,3}$	19929	10	0.67	70.61%	45.37%	51.35%	0.83	32.02%	140.87%	27.29%
$M_{1,4}$	20000	10	0.86	44.75%	1.45%	67.87%	0.75	54.03%	10.24%	67.68%
$M_{1,5}$	861906	10	0.8	19.28%	5.7%	32.7%	0.66	15.99%	50.82%	42.17%
$M_{1,6}$	1204020	10	0.8	37.22%	32.4%	22.53%	0.81	114.7%	31.4%	64.55%
$M_{2,1}$	368884	10	0.67	60.35%	32.27%	14.44%	0.65	113.24%	12.56%	2.51%
$M_{2,2}$	254907	10	0.8	11.14%	35.5%	37.77%	0.65	2.4%	30.11%	77.97%
$M_{2,3}$	16064	10	0.73	43.94%	26.25%	63.14%	0.57	13.67%	94.89%	10.95%
$M_{2,4}$	9850	10	0.77	40.07%	34.5%	10.84%	0.45	95.33%	38.07%	70.18%
$M_{2,5}$	829831	10	0.86	1.7%	13.83%	18.74%	0.72	49.81%	6.62%	0.69%
$M_{2,6}$	800249	10	0.75	34.84%	24.06%	31.16%	0.44	82.51%	29.92%	59.62%
$M_{3,1}$	336455	10	0.72	6.57%	29.76%	21.81%	0.71	44.25%	5.96%	79.15%
$M_{3,2}$	349781	10	0.74	51.21%	30.37%	23.17%	0.74	15.93%	16.71%	162.26%
$M_{3,3}$	854505	10	0.84	6.8%	21.99%	22.4%	0.67	63.54%	54.95%	80.25%
$M_{3,4}$	1061197	10	0.68	16.96%	62.63%	53.61%	0.73	75.11%	46.95%	45.01%
$M_{3,5}$	14138	10	0.83	0.94%	28.48%	7.44%	0.44	27.45%	40.88%	85.93%
$M_{3,6}$	20041	10	0.73	39.35%	14.88%	22.25%	0.61	51.53%	49.61%	16.09%
$M_{4,1}$	711713	10	0.79	26.23%	4.0%	19.54%	0.79	68.81%	172.81%	4.91%
$M_{4,2}$	331604	10	0.87	19.49%	17.38%	7.34%	0.64	94.51%	53.02%	27.09%
$M_{4,3}$	1759008	10	0.87	19.79%	22.98%	30.12%	0.68	4.55%	0.3%	14.05%
$M_{4,4}$	866703	10	0.77	26.87%	1.96%	17.93%	0.41	78.46%	41.73%	37.54%
$M_{4,5}$	20853	10	0.76	21.62%	67.41%	5.67%	0.84	32.51%	26.23%	40.37%
$M_{4,6}$	17848	10	0.85	5.29%	24.04%	32.68%	0.73	12.43%	12.09%	27.78%
$M_{5,1}$	398228	9	0.82	14.43%	28.17%	37.91%	0.58	23.51%	29.3%	75.99%
$M_{5,2}$	364948	10	0.75	39.26%	35.83%	0.04%	0.67	23.16%	11.64%	27.37%
$M_{5,3}$	751816	10	0.81	19.9%	10.25%	4.29%	0.7	30.56%	15.69%	10.51%
$M_{5,4}$	891158	10	0.76	35.1%	25.81%	16.57%	0.68	56.62%	33.53%	30.6%
$M_{5,5}$	747568	10	0.84	6.09%	5.07%	4.42%	0.71	56.34%	28.22%	19.45%
$M_{5,6}$	882872	10	0.76	10.51%	19.7%	17.3%	0.72	15.61%	33.93%	22.15%
$M_{6,1}$	773848	10	0.78	3.62%	0.08%	23.38%	0.49	70.46%	25.25%	16.39%
$M_{6,2}$	381720	10	0.77	12.76%	4.57%	48.74%	0.52	84.37%	72.91%	6.89%
$M_{6,3}$	1697377	10	0.72	18.84%	32.27%	31.09%	0.54	3.25%	46.71%	75.99%
$M_{6,4}$	953679	10	0.84	0.53%	14.57%	23.11%	0.53	91.79%	2.95%	117.07%
$M_{6,5}$	910724	10	0.8	57.73%	36.1%	3.19%	0.57	47.73%	51.16%	5.42%
$M_{6,6}$	1201092	10	0.74	21.32%	47.73%	27.47%	0.76	187.22%	34.26%	5.12%

Approximation Factor in % on the dataset email-Eu-core										
			First Variant				Second Variant			
Motif	C_M	s	$\bar{\phi}_1$	run 1	run 2	run 3	$\bar{\phi}_2$	run 1	run 2	run 3
$M_{1,1}$	514417	28	0.66	6.87%	2.28%	18.77%	0.49	54.25%	25.99%	14.04%
$M_{1,2}$	430620	29	0.63	4.92%	15.58%	0.72%	0.44	24.87%	27.57%	14.45%
$M_{1,3}$	169284	29	0.65	7.72%	7.3%	12.99%	0.55	4.55%	17.28%	23.48%
$M_{1,4}$	198631	28	0.61	19.85%	11.31%	5.29%	0.5	19.96%	3.59%	6.36%
$M_{1,5}$	626891	27	0.65	5.78%	5.25%	16.35%	0.42	12.5%	12.43%	47.87%
$M_{1,6}$	789842	28	0.62	10.72%	9.22%	16.57%	0.54	27.55%	2.92%	16.75%
$M_{2,1}$	711249	28	0.63	2.63%	0.52%	0.83%	0.52	1.11%	8.48%	23.59%
$M_{2,2}$	817579	29	0.61	15.64%	2.03%	3.46%	0.57	20.32%	14.57%	1.61%
$M_{2,3}$	160528	29	0.58	20.74%	15.24%	2.17%	0.47	45.97%	5.53%	29.78%
$M_{2,4}$	122157	29	0.65	25.67%	9.45%	15.28%	0.51	8.42%	32.84%	7.26%
$M_{2,5}$	1016020	29	0.69	4.73%	4.26%	12.01%	0.48	42.3%	12.58%	7.02%
$M_{2,6}$	626374	27	0.59	9.5%	21.16%	14.98%	0.51	3.57%	1.01%	2.66%
$M_{3,1}$	705429	29	0.61	0.49%	11.3%	17.46%	0.48	32.59%	11.29%	1.55%
$M_{3,2}$	466983	28	0.66	6.36%	2.75%	1.24%	0.45	9.62%	6.07%	9.1%
$M_{3,3}$	975941	28	0.66	15.2%	4.46%	2.94%	0.5	11.4%	13.55%	2.76%
$M_{3,4}$	1091657	29	0.6	15.57%	6.73%	10.39%	0.47	5.13%	23.59%	24.61%
$M_{3,5}$	136107	28	0.62	8.94%	9.61%	5.86%	0.44	3.9%	40.41%	16.04%
$M_{3,6}$	209354	28	0.64	10.35%	20.77%	1.06%	0.44	12.31%	3.08%	4.1%
$M_{4,1}$	3385029	28	0.69	4.16%	8.03%	2.56%	0.49	9.79%	5.4%	13.07%
$M_{4,2}$	858673	28	0.66	1.94%	11.8%	5.43%	0.46	25.03%	15.81%	64.3%
$M_{4,3}$	3693684	29	0.69	10.2%	10.54%	8.33%	0.49	1.05%	15.9%	0.99%
$M_{4,4}$	1094325	28	0.68	3.3%	6.68%	6.47%	0.41	22.21%	43.33%	27.14%
$M_{4,5}$	205742	28	0.62	19.73%	8.91%	5.51%	0.53	40.84%	19.52%	23.49%
$M_{4,6}$	207165	27	0.64	11.57%	9.02%	13.27%	0.52	0.15%	29.29%	1.31%
$M_{5,1}$	1392520	28	0.65	6.92%	36.71%	10.59%	0.45	61.06%	18.95%	10.9%
$M_{5,2}$	1362409	28	0.65	1.38%	32.51%	35.41%	0.5	20.83%	24.54%	6.21%
$M_{5,3}$	921000	28	0.61	8.36%	2.52%	4.04%	0.45	13.71%	30.22%	29.78%
$M_{5,4}$	631995	28	0.65	1.48%	0.44%	5.54%	0.49	4.65%	16.84%	14.35%
$M_{5,5}$	1072210	28	0.68	1.59%	12.27%	1.73%	0.48	21.65%	34.92%	0.4%
$M_{5,6}$	651653	29	0.64	9.08%	1.22%	29.59%	0.52	23.79%	7.78%	3.04%
$M_{6,1}$	2064940	29	0.67	11.25%	3.96%	15.88%	0.45	31.16%	6.01%	9.74%
$M_{6,2}$	1363113	29	0.63	13.56%	16.28%	13.29%	0.47	1.44%	14.34%	11.75%
$M_{6,3}$	3848912	28	0.61	21.64%	11.84%	28.49%	0.46	29.62%	21.13%	10.59%
$M_{6,4}$	1029608	28	0.65	3.98%	4.87%	5.0%	0.52	9.13%	30.33%	2.19%
$M_{6,5}$	1108292	28	0.62	15.24%	13.5%	4.74%	0.53	31.42%	10.05%	32.29%
$M_{6,6}$	810796	29	0.62	6.54%	9.77%	11.89%	0.51	15.2%	22.05%	10.61%

Approximation Factor in % on the dataset sx-SuperUser										
			First Variant				Second Variant			
Motif	C_M	s	$\bar{\phi}_1$	run 1	run 2	run 3	$\bar{\phi}_2$	run 1	run 2	run 3
$M_{1,1}$	166857	103	0.57	2.71%	6.43%	5.6%	0.52	3.75%	0.98%	2.36%
$M_{1,2}$	89735	102	0.56	3.9%	3.81%	4.34%	0.51	8.95%	0.89%	1.5%
$M_{1,3}$	31181	99	0.56	1.96%	8.56%	3.83%	0.51	12.21%	4.64%	13.12%
$M_{1,4}$	33002	99	0.54	7.51%	5.66%	8.57%	0.52	4.92%	2.45%	2.81%
$M_{1,5}$	202005	101	0.56	3.53%	0.51%	2.05%	0.51	0.68%	2.88%	8.95%
$M_{1,6}$	388398	99	0.57	4.51%	0.62%	1.93%	0.53	1.05%	6.97%	4.28%
$M_{2,1}$	132965	102	0.56	8.01%	2.35%	4.86%	0.52	6.84%	9.06%	0.56%
$M_{2,2}$	56823	101	0.57	4.31%	1.31%	1.49%	0.5	5.1%	2.3%	6.6%
$M_{2,3}$	21089	103	0.55	3.24%	2.72%	1.02%	0.5	3.5%	16.49%	17.79%
$M_{2,4}$	6334	103	0.54	7.25%	7.09%	0.5%	0.49	15.47%	10.79%	28.35%
$M_{2,5}$	326259	100	0.54	2.68%	5.18%	3.18%	0.51	3.69%	0.7%	3.89%
$M_{2,6}$	279654	99	0.55	13.04%	5.04%	3.29%	0.51	12.79%	15.29%	5.88%
$M_{3,1}$	79626	99	0.56	4.28%	0.01%	1.69%	0.53	3.52%	1.0%	1.02%
$M_{3,2}$	129020	100	0.57	4.11%	6.91%	1.66%	0.53	1.6%	9.44%	1.21%
$M_{3,3}$	147975	99	0.56	9.75%	3.26%	0.83%	0.5	16.27%	5.0%	3.64%
$M_{3,4}$	557702	102	0.55	6.79%	2.05%	3.88%	0.51	0.23%	3.09%	10.68%
$M_{3,5}$	12263	101	0.59	4.11%	1.92%	5.86%	0.53	7.66%	11.56%	8.68%
$M_{3,6}$	24870	100	0.55	3.64%	1.11%	3.76%	0.49	10.26%	4.02%	9.64%
$M_{4,1}$	233400	99	0.57	2.76%	2.17%	4.41%	0.51	4.28%	0.66%	3.02%
$M_{4,2}$	209077	102	0.57	0.15%	0.63%	1.43%	0.5	0.06%	7.65%	4.26%
$M_{4,3}$	1091788	102	0.56	3.01%	5.24%	9.47%	0.51	3.67%	2.15%	1.7%
$M_{4,4}$	601707	99	0.55	4.26%	0.31%	4.77%	0.51	0.47%	13.52%	4.09%
$M_{4,5}$	22457	100	0.55	3.86%	8.55%	3.0%	0.5	3.44%	7.74%	3.68%
$M_{4,6}$	11976	102	0.57	0.89%	3.95%	10.07%	0.51	6.02%	6.4%	1.17%
$M_{5,1}$	17898	101	0.57	3.68%	1.7%	2.06%	0.49	3.93%	1.11%	9.11%
$M_{5,2}$	112602	99	0.56	2.18%	2.26%	0.87%	0.5	0.2%	8.0%	2.51%
$M_{5,3}$	549672	101	0.57	4.27%	1.34%	9.59%	0.53	5.0%	3.41%	7.82%
$M_{5,4}$	343289	99	0.56	3.78%	5.85%	7.35%	0.52	5.87%	5.59%	5.52%
$M_{5,5}$	170469	101	0.58	0.03%	17.28%	1.42%	0.51	0.36%	9.3%	4.5%
$M_{5,6}$	184637	99	0.55	5.35%	4.02%	1.85%	0.5	18.85%	10.53%	6.71%
$M_{6,1}$	167850	99	0.55	2.29%	0.12%	4.5%	0.49	7.17%	6.05%	6.28%
$M_{6,2}$	40629	100	0.56	0.98%	2.33%	3.91%	0.5	1.07%	8.1%	6.74%
$M_{6,3}$	1059493	98	0.56	6.21%	3.86%	3.78%	0.48	3.55%	11.14%	10.97%
$M_{6,4}$	322650	98	0.56	1.42%	1.07%	1.17%	0.51	10.35%	5.0%	8.44%
$M_{6,5}$	472529	99	0.56	6.18%	1.44%	11.63%	0.51	1.33%	9.96%	2.06%
$M_{6,6}$	396247	102	0.57	0.91%	0.47%	3.94%	0.53	4.94%	3.04%	5.38%

Approximation Factor in % on the dataset FBWall

			First Variant				Second Variant			
Motif	C_M	s	$\bar{\phi}_1$	run 1	run 2	run 3	$\bar{\phi}_2$	run 1	run 2	run 3
$M_{1,1}$	115233	46	0.63	4.23%	8.26%	5.67%	0.41	23.96%	6.07%	8.12%
$M_{1,2}$	135152	46	0.65	0.72%	8.53%	0.54%	0.42	9.31%	42.35%	1.53%
$M_{1,3}$	11221	47	0.68	19.0%	9.28%	12.15%	0.39	2.12%	41.46%	53.25%
$M_{1,4}$	12256	47	0.66	3.4%	31.63%	0.88%	0.44	19.86%	20.04%	14.21%
$M_{1,5}$	251005	46	0.68	0.25%	0.52%	6.28%	0.39	21.9%	15.27%	1.91%
$M_{1,6}$	207218	46	0.63	6.41%	0.61%	2.86%	0.48	5.87%	26.71%	13.26%
$M_{2,1}$	90055	47	0.66	10.49%	7.45%	8.65%	0.41	34.66%	13.91%	9.83%
$M_{2,2}$	117750	46	0.64	2.41%	6.53%	7.83%	0.43	3.83%	8.01%	18.73%
$M_{2,3}$	14439	46	0.62	5.53%	12.46%	9.4%	0.46	13.38%	27.9%	34.33%
$M_{2,4}$	11334	47	0.67	14.44%	11.33%	16.87%	0.45	25.21%	2.5%	44.6%
$M_{2,5}$	209126	46	0.64	0.09%	1.43%	1.86%	0.45	21.28%	18.66%	17.52%
$M_{2,6}$	273644	46	0.66	0.93%	0.06%	5.12%	0.44	22.56%	0.48%	6.01%
$M_{3,1}$	106457	46	0.64	9.25%	7.04%	1.9%	0.44	21.84%	11.06%	10.86%
$M_{3,2}$	100138	46	0.64	2.44%	11.38%	2.51%	0.37	33.69%	16.9%	1.32%
$M_{3,3}$	212592	47	0.62	9.47%	7.06%	2.87%	0.51	33.58%	11.27%	7.5%
$M_{3,4}$	155864	46	0.66	19.48%	19.47%	5.63%	0.43	26.87%	6.05%	17.31%
$M_{3,5}$	7954	47	0.66	30.06%	15.4%	5.73%	0.46	76.41%	73.03%	42.47%
$M_{3,6}$	10501	46	0.68	41.0%	8.39%	22.31%	0.48	44.88%	3.19%	36.19%
$M_{4,1}$	154215	46	0.67	2.27%	3.29%	19.62%	0.45	46.22%	47.02%	34.13%
$M_{4,2}$	102091	46	0.68	0.19%	14.17%	5.15%	0.45	5.18%	0.9%	1.92%
$M_{4,3}$	265096	46	0.64	2.78%	33.82%	8.86%	0.45	11.1%	21.34%	2.35%
$M_{4,4}$	267087	46	0.6	0.01%	2.23%	6.74%	0.46	8.26%	2.24%	14.42%
$M_{4,5}$	13872	46	0.64	21.92%	11.17%	8.85%	0.43	54.33%	18.61%	22.44%
$M_{4,6}$	13074	46	0.66	10.44%	10.41%	2.91%	0.37	7.2%	60.96%	9.86%
$M_{5,1}$	970486	47	0.67	2.36%	0.57%	10.74%	0.48	0.88%	13.35%	18.68%
$M_{5,2}$	832672	47	0.64	3.5%	0.7%	18.85%	0.47	4.96%	23.92%	12.78%
$M_{5,3}$	293826	47	0.66	1.82%	10.83%	0.64%	0.41	15.74%	10.11%	4.07%
$M_{5,4}$	273445	47	0.64	6.49%	1.11%	4.27%	0.45	5.79%	15.43%	8.43%
$M_{5,5}$	243042	46	0.66	2.79%	2.68%	9.59%	0.4	0.06%	8.03%	34.99%
$M_{5,6}$	215511	47	0.64	5.47%	3.28%	3.41%	0.47	11.06%	11.5%	4.55%
$M_{6,1}$	939754	47	0.63	7.54%	7.09%	5.18%	0.49	28.37%	24.95%	5.57%
$M_{6,2}$	841299	46	0.65	15.04%	1.76%	4.5%	0.45	25.48%	30.17%	43.32%
$M_{6,3}$	270734	46	0.68	22.58%	14.39%	2.12%	0.49	54.5%	27.31%	19.17%
$M_{6,4}$	191214	47	0.63	5.62%	15.07%	5.1%	0.49	7.74%	2.21%	43.6%
$M_{6,5}$	185951	47	0.61	4.19%	8.88%	1.27%	0.48	38.25%	7.63%	5.81%
$M_{6,6}$	200650	46	0.64	5.21%	2.36%	3.71%	0.44	3.84%	27.79%	41.04%

Approximation Factor in % on the dataset SMS-ME										
			First Variant				Second Variant			
Motif	C_M	s	$\bar{\phi}_1$	run 1	run 2	run 3	$\bar{\phi}_2$	run 1	run 2	run 3
$M_{1,1}$	1011752	19	0.57	27.39%	25.82%	3.27%	0.61	25.88%	5.34%	9.76%
$M_{1,2}$	1129270	18	0.55	18.95%	57.21%	70.43%	0.64	34.62%	12.76%	3.99%
$M_{1,3}$	21587	19	0.59	35.12%	5.27%	1.33%	0.69	11.02%	23.82%	9.57%
$M_{1,4}$	22676	19	0.58	18.46%	23.83%	22.89%	0.61	6.06%	9.94%	0.73%
$M_{1,5}$	5315944	19	0.58	13.49%	58.01%	55.56%	0.65	22.71%	14.34%	20.67%
$M_{1,6}$	11469761	18	0.56	4.56%	22.27%	6.3%	0.61	14.81%	70.53%	84.12%
$M_{2,1}$	1492053	18	0.66	27.78%	59.11%	12.33%	0.59	38.77%	45.08%	52.89%
$M_{2,2}$	1705749	19	0.58	109.94%	34.48%	3.62%	0.72	5.5%	2.5%	22.35%
$M_{2,3}$	20745	19	0.59	27.96%	13.3%	5.5%	0.62	0.97%	10.57%	19.87%
$M_{2,4}$	18587	19	0.57	20.18%	12.39%	17.37%	0.65	11.2%	3.99%	3.5%
$M_{2,5}$	5945908	19	0.6	73.52%	40.14%	57.55%	0.66	8.27%	24.7%	79.99%
$M_{2,6}$	4862269	19	0.56	68.59%	19.43%	7.21%	0.69	3.59%	41.78%	42.9%
$M_{3,1}$	1404994	19	0.62	115.32%	1.25%	3.93%	0.62	50.62%	56.27%	25.9%
$M_{3,2}$	1074065	18	0.55	11.31%	59.6%	17.82%	0.7	41.83%	10.17%	40.11%
$M_{3,3}$	6053318	19	0.61	56.44%	49.06%	51.53%	0.67	69.5%	44.23%	40.43%
$M_{3,4}$	5342704	19	0.63	80.6%	45.42%	40.67%	0.64	46.37%	19.91%	10.22%
$M_{3,5}$	20847	19	0.57	41.13%	4.2%	13.03%	0.64	17.19%	2.32%	8.53%
$M_{3,6}$	24213	19	0.6	3.75%	16.76%	32.49%	0.68	6.88%	26.62%	9.36%
$M_{4,1}$	2562388	19	0.57	39.86%	22.04%	55.1%	0.61	23.78%	12.96%	58.2%
$M_{4,2}$	1656174	18	0.58	74.28%	51.13%	2.87%	0.6	13.17%	3.86%	10.86%
$M_{4,3}$	8983423	19	0.54	28.63%	42.95%	0.65%	0.62	19.59%	36.79%	61.14%
$M_{4,4}$	5739729	18	0.65	64.0%	47.2%	9.22%	0.63	35.17%	8.52%	2.99%
$M_{4,5}$	23769	18	0.59	20.02%	7.87%	27.33%	0.61	8.63%	15.16%	23.48%
$M_{4,6}$	23612	19	0.62	0.94%	8.99%	9.29%	0.62	6.97%	19.77%	0.43%
$M_{5,1}$	54657431	19	0.6	25.07%	27.45%	16.25%	0.61	44.28%	3.64%	8.89%
$M_{5,2}$	54016279	18	0.62	8.04%	17.5%	0.13%	0.67	8.23%	1.72%	5.49%
$M_{5,3}$	5527281	19	0.62	61.86%	6.27%	7.54%	0.66	10.5%	35.2%	51.97%
$M_{5,4}$	3891704	19	0.59	5.92%	23.19%	1.07%	0.67	52.82%	64.31%	73.37%
$M_{5,5}$	5113195	19	0.64	5.87%	18.7%	6.71%	0.61	5.14%	49.82%	57.18%
$M_{5,6}$	3679849	19	0.6	11.94%	56.42%	28.37%	0.59	72.54%	41.5%	57.42%
$M_{6,1}$	84304364	19	0.6	25.24%	3.75%	0.8%	0.64	9.62%	15.38%	38.97%
$M_{6,2}$	53917340	19	0.61	13.14%	3.31%	7.74%	0.63	2.8%	7.84%	6.27%
$M_{6,3}$	8277785	18	0.61	6.69%	37.85%	14.88%	0.62	23.23%	2.16%	37.64%
$M_{6,4}$	4260867	19	0.58	41.09%	21.43%	102.41%	0.67	2.37%	3.77%	65.42%
$M_{6,5}$	4910880	19	0.6	73.61%	57.99%	58.22%	0.63	8.09%	18.18%	68.06%
$M_{6,6}$	4858353	19	0.54	71.32%	37.3%	52.79%	0.66	7.02%	23.25%	85.4%

Approximation Factor in % on the dataset MathOverflow

			First Variant				Second Variant			
Motif	C_M	s	$\bar{\phi}_1$	run 1	run 2	run 3	$\bar{\phi}_2$	run 1	run 2	run 3
$M_{1,1}$	131573	94	0.56	0.47%	4.88%	0.98%	0.56	2.51%	2.88%	6.37%
$M_{1,2}$	44890	94	0.56	8.43%	3.42%	4.99%	0.55	2.87%	1.74%	7.42%
$M_{1,3}$	26091	94	0.56	0.44%	6.44%	15.05%	0.54	9.07%	5.65%	5.97%
$M_{1,4}$	21087	92	0.55	2.11%	10.96%	7.78%	0.53	2.79%	11.71%	24.28%
$M_{1,5}$	99043	93	0.55	0.84%	3.6%	4.14%	0.55	3.54%	6.91%	1.57%
$M_{1,6}$	217732	94	0.57	8.19%	1.61%	1.88%	0.58	4.06%	7.3%	9.53%
$M_{2,1}$	34272	93	0.55	17.33%	18.75%	2.5%	0.55	7.8%	10.93%	17.09%
$M_{2,2}$	22100	92	0.55	3.03%	8.39%	19.74%	0.55	2.89%	0.95%	8.14%
$M_{2,3}$	13368	94	0.54	12.0%	13.96%	17.38%	0.52	9.75%	2.51%	20.92%
$M_{2,4}$	3653	94	0.54	1.52%	13.91%	9.62%	0.55	1.92%	8.06%	6.35%
$M_{2,5}$	101393	94	0.57	1.55%	4.71%	0.64%	0.52	5.46%	4.66%	2.42%
$M_{2,6}$	40368	92	0.58	6.61%	0.65%	9.04%	0.54	8.0%	28.97%	3.1%
$M_{3,1}$	34576	90	0.57	12.9%	3.72%	13.71%	0.55	22.17%	7.76%	1.14%
$M_{3,2}$	41057	93	0.57	2.29%	8.41%	7.25%	0.53	9.82%	8.39%	0.73%
$M_{3,3}$	52813	94	0.58	20.67%	1.82%	2.06%	0.54	3.01%	5.71%	6.48%
$M_{3,4}$	116977	94	0.55	0.26%	4.5%	15.16%	0.52	4.11%	0.29%	5.61%
$M_{3,5}$	8247	93	0.57	3.59%	4.0%	0.33%	0.55	4.62%	18.83%	16.65%
$M_{3,6}$	16182	94	0.54	5.23%	1.51%	8.87%	0.56	18.1%	3.63%	9.86%
$M_{4,1}$	54579	93	0.55	1.52%	3.71%	0.9%	0.54	2.19%	14.16%	10.07%
$M_{4,2}$	29390	94	0.55	16.36%	13.21%	10.5%	0.53	3.47%	1.6%	22.56%
$M_{4,3}$	163441	93	0.55	1.63%	8.88%	6.14%	0.53	0.54%	8.97%	14.77%
$M_{4,4}$	56309	94	0.55	5.1%	12.31%	0.02%	0.54	21.98%	2.7%	15.88%
$M_{4,5}$	14552	92	0.56	5.62%	5.9%	15.72%	0.55	8.81%	3.62%	4.57%
$M_{4,6}$	7518	93	0.58	11.3%	5.66%	2.91%	0.54	4.75%	0.78%	8.16%
$M_{5,1}$	9184	93	0.53	1.31%	14.59%	18.13%	0.54	2.78%	8.49%	5.41%
$M_{5,2}$	32226	94	0.58	1.54%	4.12%	0.22%	0.52	4.65%	0.05%	3.56%
$M_{5,3}$	54195	94	0.56	14.94%	12.19%	18.74%	0.54	0.36%	13.82%	1.66%
$M_{5,4}$	62683	94	0.54	5.69%	4.7%	11.77%	0.53	3.34%	15.12%	5.59%
$M_{5,5}$	56811	90	0.55	14.45%	8.73%	9.42%	0.54	3.5%	12.81%	21.57%
$M_{5,6}$	78042	93	0.55	3.69%	0.29%	7.47%	0.55	2.38%	5.16%	4.38%
$M_{6,1}$	91919	94	0.57	15.24%	1.98%	19.64%	0.55	11.97%	14.15%	11.11%
$M_{6,2}$	18407	93	0.56	12.8%	10.88%	3.82%	0.53	0.55%	1.51%	1.69%
$M_{6,3}$	163078	92	0.55	5.27%	6.01%	9.64%	0.53	4.06%	1.28%	8.19%
$M_{6,4}$	108044	91	0.55	3.74%	6.59%	4.89%	0.53	1.22%	13.44%	1.99%
$M_{6,5}$	96115	94	0.56	0.74%	11.58%	6.49%	0.55	0.92%	9.95%	14.42%
$M_{6,6}$	217864	94	0.56	1.2%	2.39%	1.64%	0.55	5.43%	4.03%	0.8%

Approximation Factor in % on the dataset AskUbuntu										
			First Variant				Second Variant			
Motif	C_M	s	$\bar{\phi}_1$	run 1	run 2	run 3	$\bar{\phi}_2$	run 1	run 2	run 3
$M_{1,1}$	126939	89	0.6	3.76%	7.77%	0.13%	0.52	16.02%	8.76%	2.64%
$M_{1,2}$	63189	88	0.62	4.26%	2.19%	3.16%	0.49	8.24%	2.35%	4.51%
$M_{1,3}$	19951	89	0.61	6.83%	11.89%	2.21%	0.47	2.16%	6.66%	4.47%
$M_{1,4}$	19804	89	0.62	9.23%	0.64%	9.27%	0.49	2.0%	0.35%	2.58%
$M_{1,5}$	168096	88	0.59	0.13%	2.06%	3.16%	0.49	16.12%	1.63%	15.18%
$M_{1,6}$	426148	89	0.59	5.24%	4.92%	6.75%	0.47	5.46%	15.3%	7.08%
$M_{2,1}$	139655	88	0.6	3.4%	0.81%	3.89%	0.54	3.95%	34.85%	7.23%
$M_{2,2}$	55458	89	0.6	3.5%	16.98%	2.91%	0.52	13.31%	13.44%	3.81%
$M_{2,3}$	13713	88	0.6	4.84%	4.22%	7.66%	0.52	21.9%	7.96%	6.7%
$M_{2,4}$	5509	88	0.58	6.88%	19.82%	5.02%	0.51	18.0%	19.16%	10.26%
$M_{2,5}$	413142	89	0.57	0.79%	0.14%	15.23%	0.5	12.03%	26.02%	4.74%
$M_{2,6}$	298509	89	0.6	3.67%	2.7%	18.74%	0.52	2.33%	25.18%	1.69%
$M_{3,1}$	79470	88	0.6	0.9%	3.45%	7.54%	0.48	13.15%	9.32%	2.26%
$M_{3,2}$	113270	89	0.6	7.64%	9.13%	1.72%	0.49	1.49%	7.08%	0.48%
$M_{3,3}$	179073	89	0.62	7.36%	4.5%	8.11%	0.46	13.33%	14.51%	4.64%
$M_{3,4}$	745480	90	0.6	8.03%	4.25%	4.56%	0.51	7.57%	2.7%	0.02%
$M_{3,5}$	8946	89	0.6	12.52%	0.4%	7.14%	0.47	0.08%	11.91%	9.31%
$M_{3,6}$	14639	89	0.61	6.35%	10.75%	4.12%	0.47	2.85%	0.9%	12.34%
$M_{4,1}$	199595	89	0.61	7.66%	0.0%	1.75%	0.43	17.75%	22.54%	3.47%
$M_{4,2}$	207846	89	0.6	0.08%	6.59%	7.73%	0.5	3.05%	11.24%	3.62%
$M_{4,3}$	1044513	89	0.58	5.95%	4.79%	6.99%	0.44	23.24%	16.48%	0.2%
$M_{4,4}$	660015	89	0.61	10.33%	2.41%	7.4%	0.5	3.1%	7.44%	2.8%
$M_{4,5}$	15512	88	0.61	6.03%	9.27%	2.22%	0.49	14.16%	11.56%	6.43%
$M_{4,6}$	8346	89	0.61	16.31%	20.01%	8.85%	0.48	9.99%	5.88%	0.98%
$M_{5,1}$	21737	88	0.59	5.73%	12.27%	0.65%	0.46	2.15%	2.72%	0.77%
$M_{5,2}$	133038	89	0.6	2.34%	0.33%	2.17%	0.54	8.49%	27.7%	12.03%
$M_{5,3}$	613592	89	0.59	6.36%	14.45%	0.12%	0.47	4.12%	2.55%	10.5%
$M_{5,4}$	344810	89	0.57	6.54%	6.98%	5.1%	0.51	4.46%	9.06%	9.38%
$M_{5,5}$	184475	88	0.6	9.15%	11.0%	0.87%	0.51	4.11%	6.76%	7.97%
$M_{5,6}$	152517	88	0.6	0.92%	5.41%	0.54%	0.47	12.92%	10.29%	11.27%
$M_{6,1}$	248091	89	0.62	3.85%	8.0%	5.5%	0.51	14.49%	0.89%	10.61%
$M_{6,2}$	54097	88	0.61	5.39%	1.76%	4.52%	0.51	13.45%	4.4%	1.83%
$M_{6,3}$	994099	88	0.6	1.93%	5.17%	5.06%	0.5	17.75%	6.97%	0.79%
$M_{6,4}$	410366	90	0.61	9.77%	4.65%	2.48%	0.51	18.86%	2.97%	13.11%
$M_{6,5}$	654357	88	0.6	3.11%	1.1%	4.01%	0.49	10.61%	10.52%	9.61%
$M_{6,6}$	438895	88	0.61	3.72%	4.05%	2.73%	0.47	10.33%	12.13%	4.68%

Approximation Factor in % on the dataset WikiTalk

			First Variant				Second Variant			
Motif	C_M	s	$\bar{\phi}_1$	run 1	run 2	run 3	$\bar{\phi}_2$	run 1	run 2	run 3
$M_{1,1}$	539857	103	0.08	0.43%	0.6%	9.92%	0.04	28.43%	15.39%	20.19%
$M_{1,2}$	135226	102	0.08	4.96%	3.99%	2.56%	0.03	4.08%	1.3%	18.33%
$M_{1,3}$	86836	103	0.08	5.02%	12.55%	10.58%	0.03	5.98%	12.97%	24.29%
$M_{1,4}$	100824	103	0.08	1.52%	6.85%	14.34%	0.04	12.55%	10.78%	2.43%
$M_{1,5}$	266910	102	0.08	1.82%	8.77%	1.38%	0.03	29.72%	5.41%	10.21%
$M_{1,6}$	932924	100	0.08	27.24%	38.92%	0.72%	0.03	33.78%	22.23%	13.93%
$M_{2,1}$	446410	99	0.08	126.37%	19.95%	13.46%	0.03	44.54%	33.2%	18.38%
$M_{2,2}$	248332	102	0.09	6.96%	9.03%	53.59%	0.03	5.32%	32.86%	3.18%
$M_{2,3}$	56354	102	0.08	0.94%	4.8%	11.33%	0.04	5.55%	14.08%	9.15%
$M_{2,4}$	21941	103	0.08	12.73%	1.75%	5.18%	0.03	29.86%	7.3%	14.31%
$M_{2,5}$	1017969	100	0.08	19.83%	13.67%	16.17%	0.03	10.63%	104.09%	47.67%
$M_{2,6}$	229169	99	0.08	8.73%	34.44%	7.19%	0.03	10.95%	22.81%	23.23%
$M_{3,1}$	462985	103	0.08	30.8%	23.61%	19.23%	0.03	46.28%	91.12%	46.0%
$M_{3,2}$	158717	104	0.08	0.37%	1.46%	11.74%	0.03	37.3%	13.12%	19.02%
$M_{3,3}$	831400	103	0.08	17.69%	0.27%	16.01%	0.03	17.7%	3.8%	7.04%
$M_{3,4}$	1437961	101	0.08	37.41%	100.69%	28.74%	0.03	37.09%	48.45%	46.32%
$M_{3,5}$	31988	103	0.08	2.27%	14.7%	17.46%	0.04	4.85%	39.58%	62.6%
$M_{3,6}$	78717	101	0.08	5.44%	0.74%	1.25%	0.03	17.41%	25.31%	8.04%
$M_{4,1}$	176665319	101	0.08	37.34%	39.08%	12.01%	0.03	99.14%	45.32%	96.65%
$M_{4,2}$	579299	102	0.08	10.53%	11.07%	1.72%	0.04	14.53%	31.46%	2.07%
$M_{4,3}$	376601375	102	0.08	36.58%	41.91%	130.76%	0.03	94.67%	71.68%	190.88%
$M_{4,4}$	991003	101	0.09	9.92%	3.73%	1.96%	0.03	27.57%	14.94%	25.2%
$M_{4,5}$	-	-	-	-	-	-	-	-	-	-
$M_{4,6}$	-	-	-	-	-	-	-	-	-	-
$M_{5,1}$	918754	103	0.08	3.24%	10.16%	3.85%	0.03	7.35%	6.49%	0.55%
$M_{5,2}$	825696	102	0.08	8.98%	10.55%	2.2%	0.03	136.81%	0.85%	159.15%
$M_{5,3}$	713196	100	0.08	2.53%	15.82%	37.69%	0.03	14.97%	24.62%	12.91%
$M_{5,4}$	305617	104	0.09	1.8%	18.06%	5.16%	0.03	5.15%	7.32%	18.03%
$M_{5,5}$	655935	101	0.09	0.72%	8.84%	6.38%	0.04	27.51%	89.65%	14.65%
$M_{5,6}$	314878	101	0.08	4.0%	2.55%	2.25%	0.03	3.69%	10.27%	12.23%
$M_{6,1}$	15047345	100	0.08	43.99%	19.08%	16.79%	0.03	56.56%	1.81%	41.22%
$M_{6,2}$	799080	102	0.08	8.38%	19.34%	5.22%	0.03	16.48%	11.4%	18.57%
$M_{6,3}$	382034965	101	0.09	2.11%	196.59%	7.47%	0.03	74.44%	79.42%	66.02%
$M_{6,4}$	1538217	101	0.08	32.28%	39.48%	1.68%	0.03	118.15%	59.44%	55.32%
$M_{6,5}$	910740	101	0.08	57.45%	27.82%	23.36%	0.03	41.29%	45.96%	50.88%
$M_{6,6}$	972272	102	0.08	22.0%	13.6%	6.35%	0.03	36.23%	9.03%	64.89%

5.3 Running time comparison of the sequential methods

In this section we report the results of the running time comparison between the different methods we considered. We compared the running times between the exact routine of Machey et al. implemented by Liu et al., the algorithm BT+S with $r = 100$, the algorithm BT+S with $r = 30$, our first algorithm which we called V1, and our improved algorithm which we called V2. The results on the different datasets are reported from figure 5.1 to figure 5.8. In such plots we have for each dataset on the x axis the 36 motifs, and on the y axis the running time in seconds, such value is obtained from the geometric mean of the three runs on each motif for each configuration, namely let t_1, t_2, t_3 be the running times for run 1, run 2, run 3 once fixed the algorithm and the motif, then the time in the plot is obtained through $\sqrt[3]{\prod_i t_i}$

For the BT+S algorithms we can see that, as already anticipated, on the datasets where $\varphi_1 \sim \varphi_2$ then the two configurations of BT+S have essentially the same running times, while for example in the dataset Askubuntu where the difference between the two φ is non negligible also the difference in time is much more visible. We also observe that when $\varphi \sim 1$ then the running time of the BT+S algorithm tends to be similar or sometimes even greater than the running time of the exact routine.

For our algorithms, we can see that the first version usually requires much more time than the second one, even if the number of samples s is the same for the two methods. This is not surprising, since the computation of r_U in the first variant may require much more time w.r.t. the computation of $\tilde{r}_U$ in the second variant. Comparing instead our algorithms with the two versions of BT+S, we observe that V2 has usually a lower running time than the BT+S with $r = 100$ while our first variant has similar running times to BT+S with $r = 100$. We also observe that the datasets are quite small (see table 5.1), so it is interesting to look at the running times on the wikitalk dataset which is our biggest dataset. On such dataset, the exact routine requires much more time than the sampling based algorithms; which motivates the requirements of scalable and efficient sampling algorithms. We also recall that the approximation factor on such dataset is quite high, both for the BT+S routines and ours, which is due to the fact that few samples are used ($\varphi << 1$), to obtain a better approximation it is thus necessary to increase the running time, processing a larger number of samples.

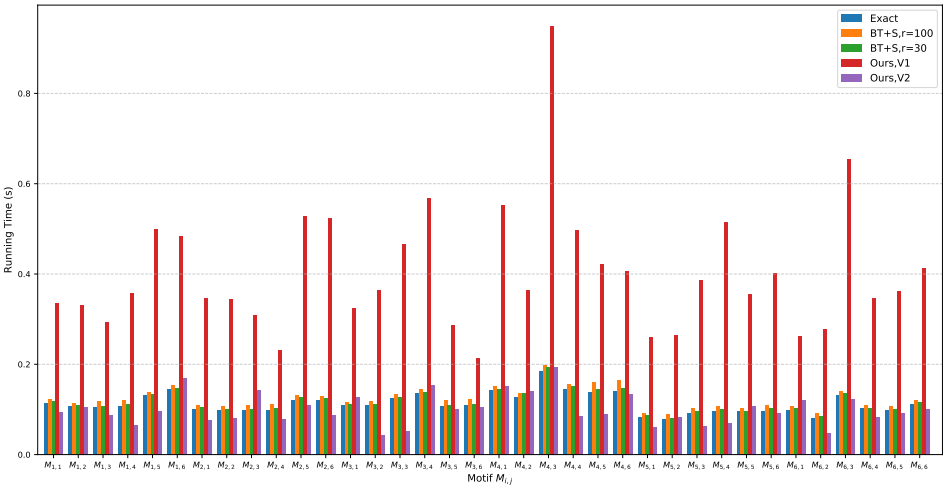


Figure 5.1: Running times for each motif on **CollegeMsg** dataset.

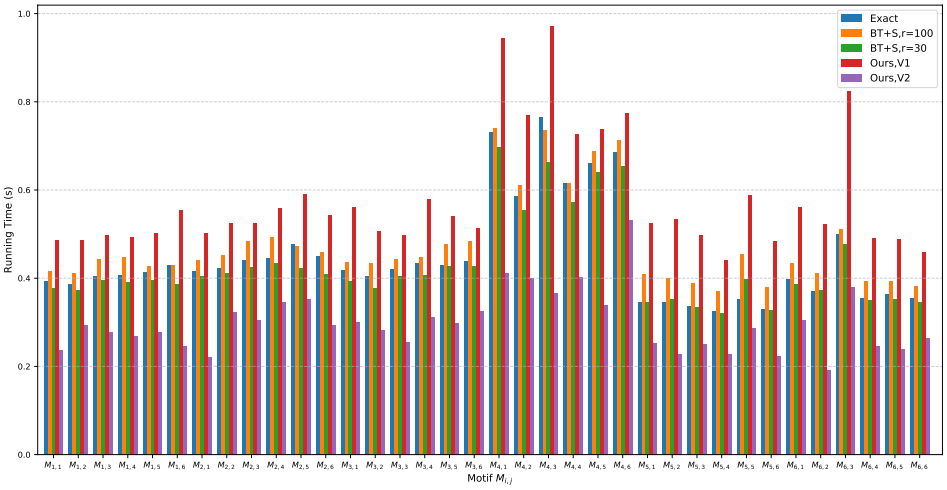


Figure 5.2: Running times for each motif on **email-Eu-core** dataset.

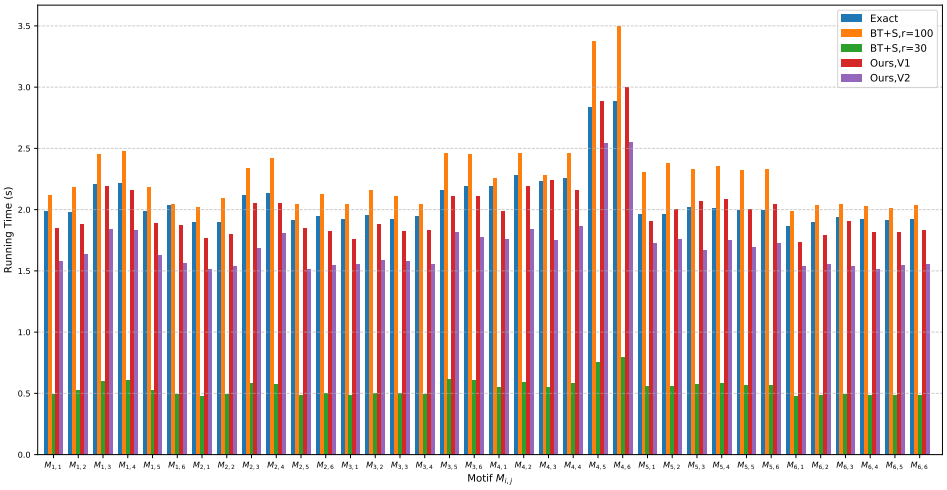


Figure 5.3: Running times for each motif on **sx-SuperUser** dataset.

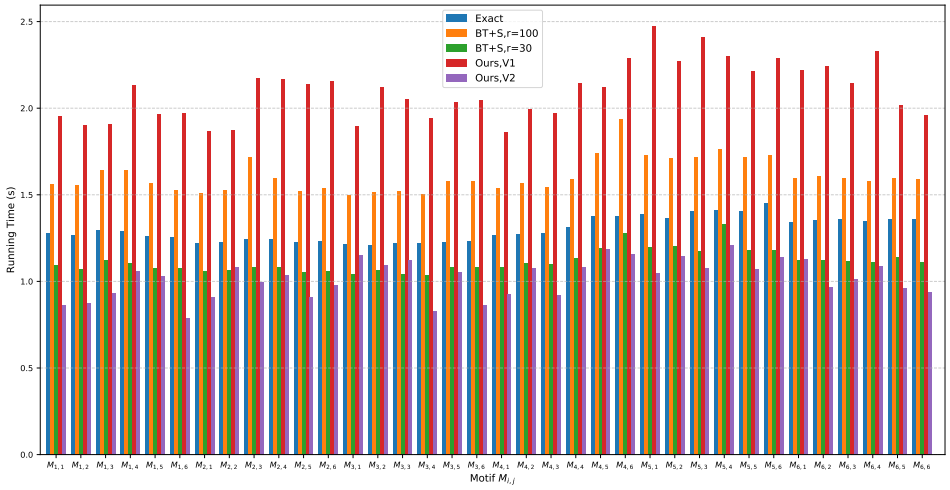


Figure 5.4: Running times for each motif on **FBWall** dataset.

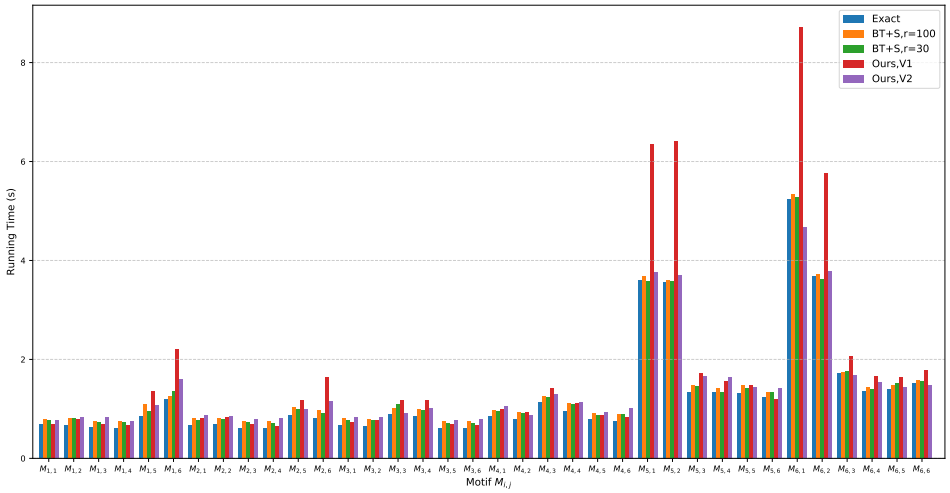


Figure 5.5: Running times for each motif on **SMS-ME** dataset.

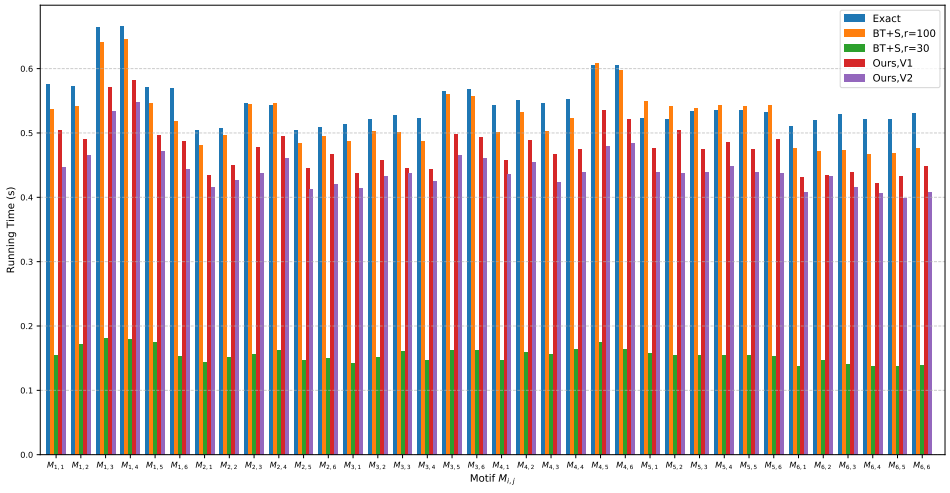


Figure 5.6: Running times for each motif on **MathOverflow** dataset.

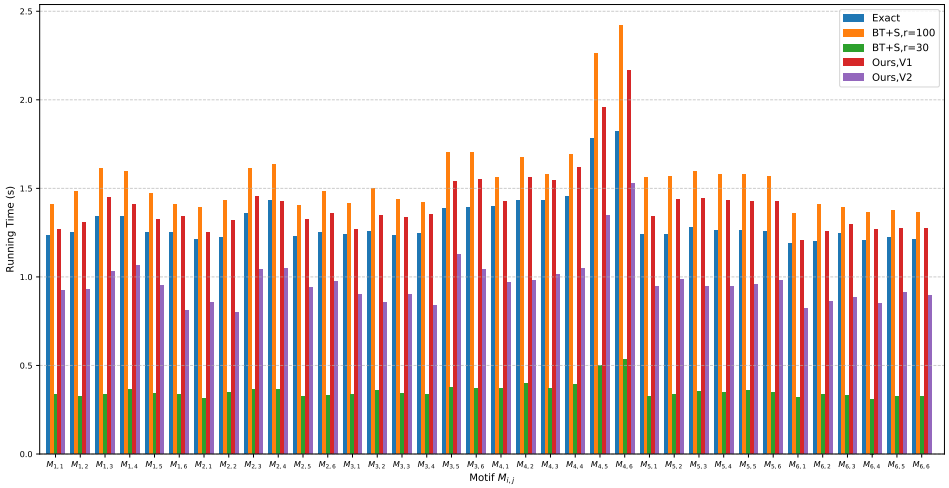


Figure 5.7: Running times for each motif on **AskUbuntu** dataset.

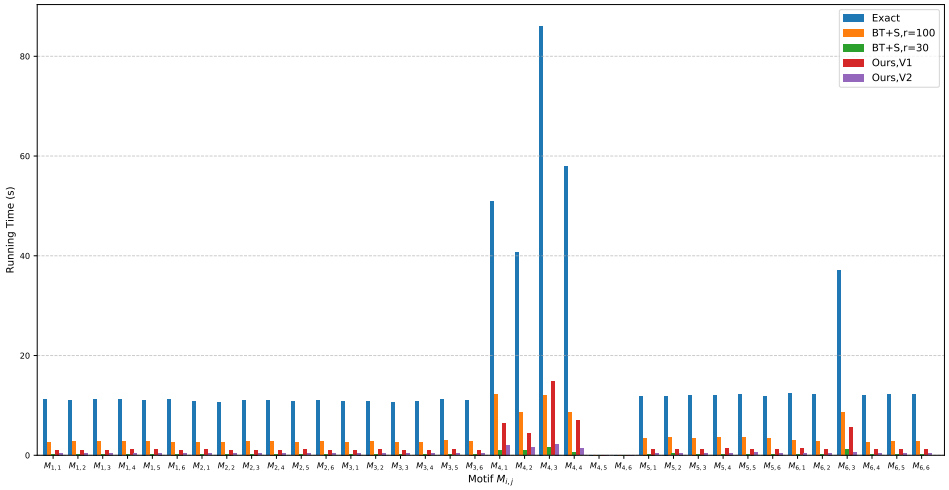


Figure 5.8: Running times for each motif on **Wikitalk** dataset.

5.4 Approximation factor on wikitalk of the parallel approaches

We implemented our algorithms also in a parallel version which gave us the possibility to compare our algorithms with the algorithm BT+PS of Liu et al., which is a parallel version of the BT+S algorithm. We also implemented our parallel-exact procedure which can be used to compute C_M in parallel (see Chapter 4). Thanks to these ingredients we were able to execute the tests on the dataset wikitalk of such routines with $\delta = 86400$. The results are shown in the tables at pages 68 and 69, we can see that the same observations we made until this moment still hold, in particular BT+PS with $r = 100$ performs better than it's version with $r = 30$ and our techniques have a worse approximation factor than the techniques of Liu et al. Such result is not surprising but is interesting compared with the previous results on the same dataset with $\delta = 3200$, where our algorithms have similar, sometimes even better, approximation factors w.r.t. the methods of Liu et al.

Approximation Factor in % on the dataset WikiTalk											
		BT+PS - $r = 100$					BT+PS - $r = 30$				
Motif	C_M	$\bar{s}_1$	φ_1	run 1	run 2	run 3	$\bar{s}_2$	φ_2	run 1	run 2	run 3
$M_{1,1}$	8467194	49	0.94	2.05%	2.45%	2.65%	29	0.67	5.75%	5.93%	5.92%
$M_{1,2}$	2115630	49	0.94	1.28%	1.1%	1.1%	31	0.71	2.81%	2.32%	3.11%
$M_{1,3}$	927677	49	0.94	1.29%	2.96%	3.0%	32	0.73	4.55%	1.37%	10.05%
$M_{1,4}$	961976	50	0.94	2.14%	2.73%	0.37%	30	0.7	1.73%	1.81%	1.66%
$M_{1,5}$	6223045	49	0.94	0.82%	1.47%	0.15%	30	0.68	1.82%	0.21%	0.23%
$M_{1,6}$	18675425	49	0.94	2.13%	1.87%	1.16%	32	0.73	5.64%	5.92%	11.78%
$M_{2,1}$	14747039	49	0.94	0.58%	0.34%	0.95%	32	0.73	0.74%	10.2%	6.6%
$M_{2,2}$	6604184	49	0.94	0.26%	0.89%	0.45%	32	0.73	8.74%	0.29%	0.41%
$M_{2,3}$	723812	50	0.94	1.68%	0.11%	1.59%	30	0.67	4.9%	4.88%	0.99%
$M_{2,4}$	330579	49	0.94	0.33%	1.47%	2.25%	32	0.73	1.52%	9.89%	9.92%
$M_{2,5}$	32488474	49	0.94	0.95%	0.1%	0.87%	30	0.68	3.62%	0.18%	2.66%
$M_{2,6}$	5821053	49	0.94	0.49%	0.39%	0.15%	31	0.71	1.2%	3.89%	2.14%
$M_{3,1}$	15149900	50	0.94	0.97%	0.61%	0.85%	31	0.71	3.42%	3.79%	3.93%
$M_{3,2}$	2678302	49	0.94	1.15%	0.82%	0.89%	31	0.71	8.0%	3.28%	4.05%
$M_{3,3}$	14898726	49	0.94	1.39%	0.18%	10.37%	32	0.73	0.14%	0.61%	7.27%
$M_{3,4}$	37110092	49	0.94	0.34%	0.81%	0.55%	32	0.73	1.09%	9.78%	1.37%
$M_{3,5}$	478996	50	0.94	1.12%	2.74%	2.32%	31	0.71	4.59%	4.23%	3.37%
$M_{3,6}$	917969	50	0.94	2.12%	0.12%	0.83%	30	0.68	1.29%	1.21%	0.71%
$M_{4,1}$	1936421730	49	0.94	1.2%	0.48%	1.0%	30	0.67	12.67%	14.64%	13.33%
$M_{4,2}$	9559467	49	0.94	3.33%	0.27%	0.27%	30	0.67	2.59%	2.64%	2.72%
$M_{4,3}$	3250417677	49	0.94	3.9%	1.36%	0.91%	31	0.71	6.26%	10.95%	0.65%
$M_{4,4}$	18307141	49	0.94	0.11%	0.39%	1.0%	30	0.68	0.05%	1.05%	2.51%
$M_{4,5}$	-	-	-	-	-	-	-	-	-	-	-
$M_{4,6}$	-	-	-	-	-	-	-	-	-	-	-
$M_{5,1}$	4498052	49	0.94	0.21%	0.18%	0.28%	32	0.73	0.62%	2.23%	6.52%
$M_{5,2}$	4550745	48	0.94	0.39%	0.96%	0.4%	30	0.67	0.92%	2.53%	2.45%
$M_{5,3}$	16434588	49	0.94	2.59%	0.91%	1.74%	30	0.67	4.42%	2.02%	5.84%
$M_{5,4}$	6652470	50	0.94	0.41%	2.41%	0.42%	30	0.69	3.21%	3.11%	3.16%
$M_{5,5}$	15620900	49	0.94	0.42%	0.85%	0.95%	30	0.68	3.77%	1.24%	3.85%
$M_{5,6}$	6858394	49	0.94	3.19%	1.26%	1.6%	32	0.73	3.96%	6.86%	0.63%
$M_{6,1}$	42572061	50	0.94	0.02%	0.87%	0.11%	30	0.67	4.41%	9.2%	4.48%
$M_{6,2}$	4385596	49	0.94	0.51%	0.9%	1.61%	30	0.68	0.58%	2.88%	0.44%
$M_{6,3}$	3416747081	49	0.94	0.7%	0.01%	0.72%	30	0.68	1.16%	11.1%	13.75%
$M_{6,4}$	48528243	50	0.94	0.28%	0.39%	0.05%	31	0.71	18.05%	7.1%	3.6%
$M_{6,5}$	28003470	49	0.94	0.59%	0.68%	0.52%	32	0.73	2.98%	8.48%	1.82%
$M_{6,6}$	18879959	50	0.94	0.6%	3.18%	2.96%	31	0.71	3.66%	3.97%	7.95%

Approximation Factor in % on the dataset WikiTalk								
			First Variant			Second Variant		
Motif	C_M	s	run 1	run 2	run 3	run 1	run 2	run 3
$M_{1,1}$	8467194	49	2.72%	6.31%	5.83%	19.84%	11.97%	15.45%
$M_{1,2}$	2115630	49	1.05%	0.2%	1.93%	4.56%	13.37%	10.25%
$M_{1,3}$	927677	49	1.92%	0.58%	1.69%	31.42%	10.07%	24.43%
$M_{1,4}$	961976	50	0.48%	4.12%	7.89%	0.79%	4.72%	22.44%
$M_{1,5}$	6223045	49	1.68%	4.2%	4.7%	10.81%	18.1%	29.85%
$M_{1,6}$	18675425	49	6.91%	14.62%	5.31%	11.83%	12.28%	14.91%
$M_{2,1}$	14747039	49	58.28%	58.09%	7.22%	75.07%	46.34%	54.51%
$M_{2,2}$	6604184	49	6.19%	1.88%	19.68%	51.66%	37.22%	59.57%
$M_{2,3}$	723812	50	6.22%	1.14%	1.16%	33.98%	9.88%	23.96%
$M_{2,4}$	330579	49	1.01%	5.39%	6.14%	15.73%	35.74%	11.52%
$M_{2,5}$	32488474	49	24.38%	50.41%	2.38%	58.24%	45.71%	65.81%
$M_{2,6}$	5821053	49	3.06%	0.43%	3.15%	9.27%	36.37%	8.79%
$M_{3,1}$	15149900	50	32.22%	1.73%	26.77%	225.15%	32.58%	74.58%
$M_{3,2}$	2678302	49	1.85%	3.03%	0.09%	10.97%	11.15%	13.18%
$M_{3,3}$	14898726	49	15.21%	12.77%	18.94%	31.24%	41.23%	54.89%
$M_{3,4}$	37110092	49	82.88%	21.19%	0.15%	45.3%	54.89%	32.13%
$M_{3,5}$	478996	50	3.55%	2.74%	2.06%	17.34%	14.81%	12.3%
$M_{3,6}$	917969	50	3.03%	1.9%	1.57%	7.75%	29.93%	26.3%
$M_{4,1}$	1936421730	49	40.95%	32.45%	15.7%	50.54%	0.22%	7.0%
$M_{4,2}$	9559467	49	14.39%	20.69%	37.64%	47.08%	44.68%	183.06%
$M_{4,3}$	3250417677	49	29.54%	21.69%	49.78%	35.08%	25.73%	64.53%
$M_{4,4}$	18307141	49	32.94%	14.79%	12.62%	27.19%	20.39%	18.55%
$M_{4,5}$	-	-	-	-	-	-	-	-
$M_{4,6}$	-	-	-	-	-	-	-	-
$M_{5,1}$	4498052	49	6.91%	4.04%	1.76%	12.99%	30.13%	40.76%
$M_{5,2}$	4550745	48	14.0%	7.19%	1.51%	44.06%	13.1%	53.67%
$M_{5,3}$	16434588	49	3.34%	27.26%	3.25%	46.78%	42.71%	19.85%
$M_{5,4}$	6652470	50	1.21%	6.06%	5.28%	10.0%	3.12%	13.23%
$M_{5,5}$	15620900	49	12.6%	9.07%	12.98%	28.62%	49.3%	42.18%
$M_{5,6}$	6858394	49	3.96%	1.54%	5.31%	1.65%	42.41%	9.7%
$M_{6,1}$	42572061	50	32.92%	25.67%	24.72%	16.09%	16.39%	19.47%
$M_{6,2}$	4385596	49	2.92%	18.45%	14.79%	11.38%	28.68%	3.86%
$M_{6,3}$	3416747081	49	24.73%	14.6%	4.75%	76.05%	74.19%	5.93%
$M_{6,4}$	48528243	50	23.34%	49.88%	28.11%	13.3%	41.42%	56.89%
$M_{6,5}$	28003470	49	10.11%	23.73%	4.54%	39.64%	33.93%	147.89%
$M_{6,6}$	18879959	50	3.92%	3.45%	12.4%	7.19%	30.86%	43.36%

5.5 Running time comparison of the parallel methods

In this section we conclude comparing the running times of the parallel approaches. We compared our parallel-exact approach with the two versions of BT+PS with $r = 100, r = 30$ of Liu et al., and our first version with parallel sampling (V1+PS) and the improved version with parallel sampling (V2+PS). The comparison is shown from figure 5.9 to figure 5.16.

All the considerations made for the sequential algorithms still hold for their parallel implementation for the BT+PS, V1+PS and V2+PS. In particular the version of BT+PS with $r = 100$ is much slower than the one with $r = 30$, V2+PS is usually much faster than the BT+PS,$r = 100$ and the V1+PS sometimes is much slower than the other sampling algorithms. We observe that all the parallel routines improve their sequential versions. It is also interesting to note that, our parallel-exact version has very good performances on small datasets and quite good performances on the dataset wikitalk, recall that such routine is exact and in our tests results thus also efficient and scalable, as we can see from the plots.

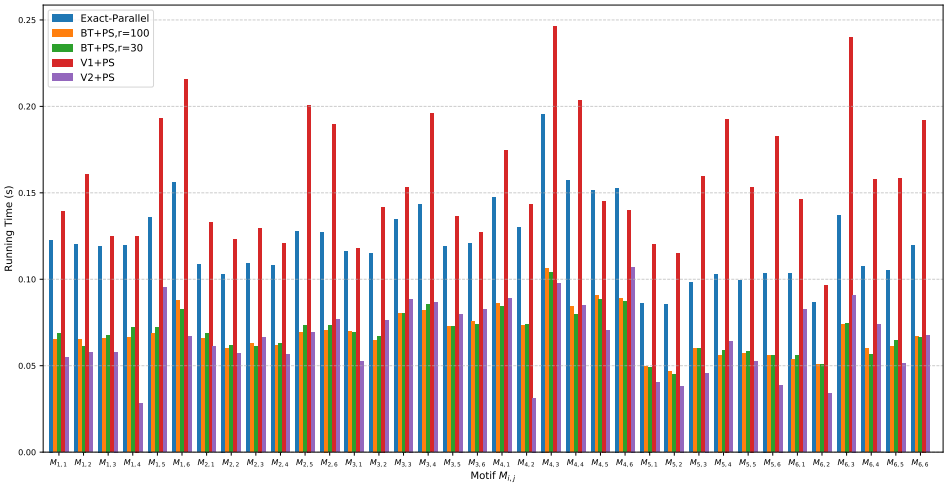


Figure 5.9: Running times for each motif on **CollegeMsg** dataset.

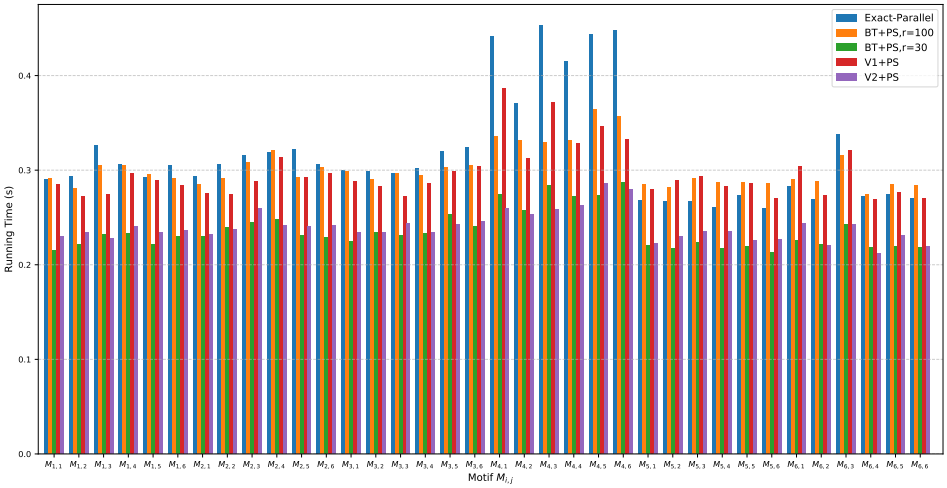


Figure 5.10: Running times for each motif on **email-Eu-core** dataset.

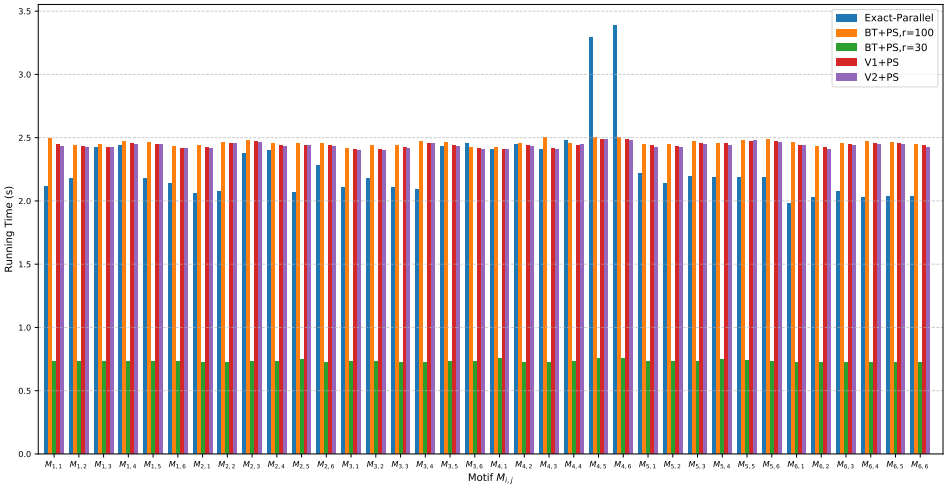


Figure 5.11: Running times for each motif on **sx-SuperUser** dataset.

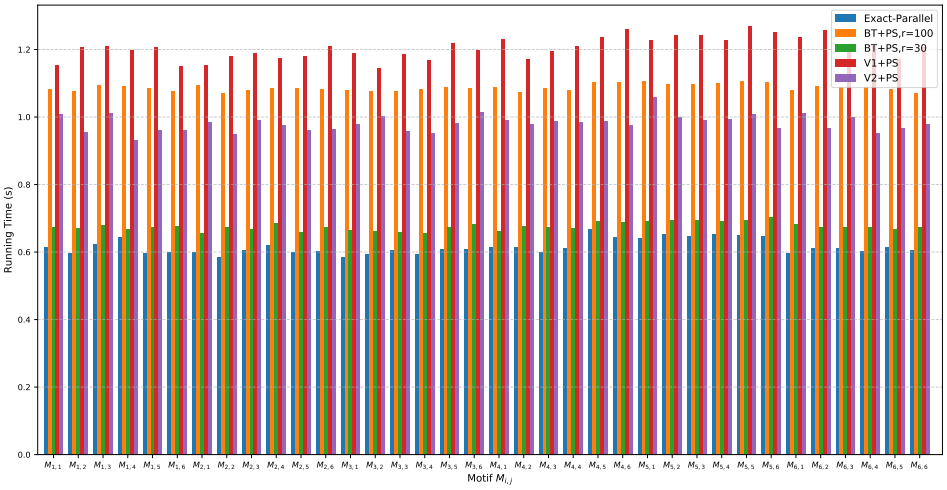


Figure 5.12: Running times for each motif on **FBWall** dataset.

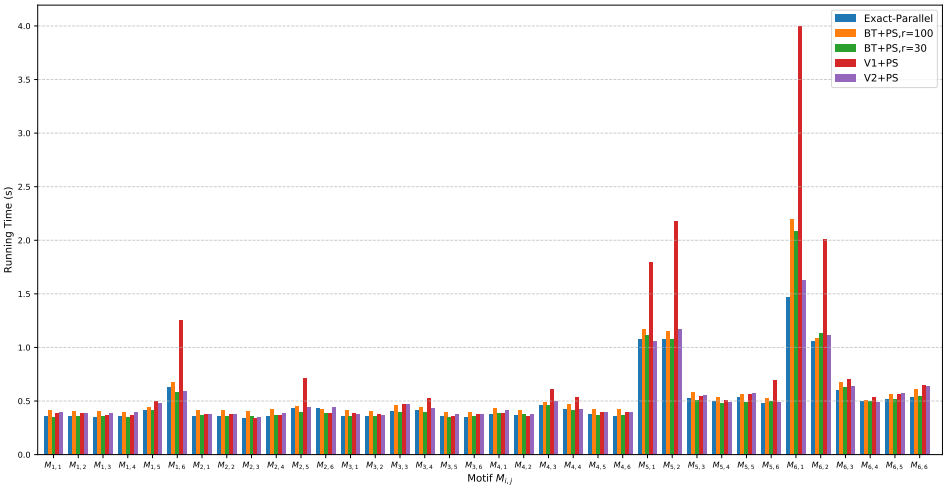


Figure 5.13: Running times for each motif on **SMS-ME** dataset.

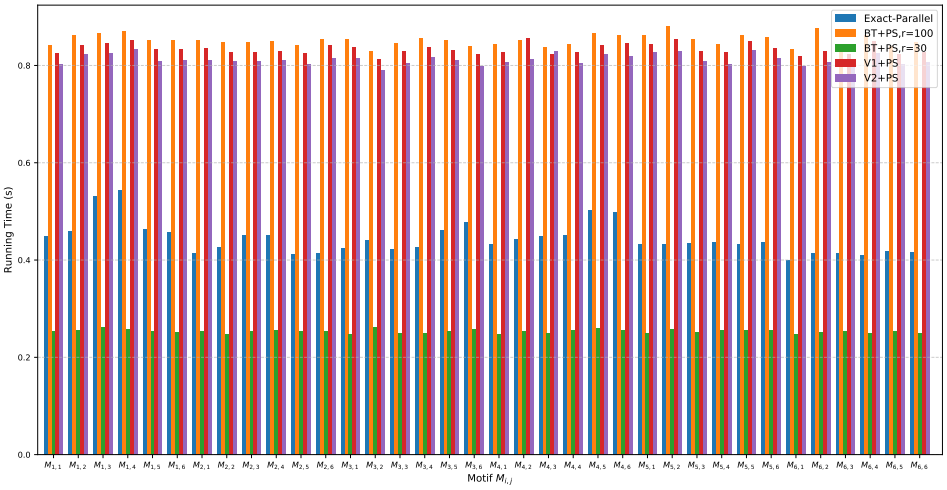


Figure 5.14: Running times for each motif on **MathOverflow** dataset.

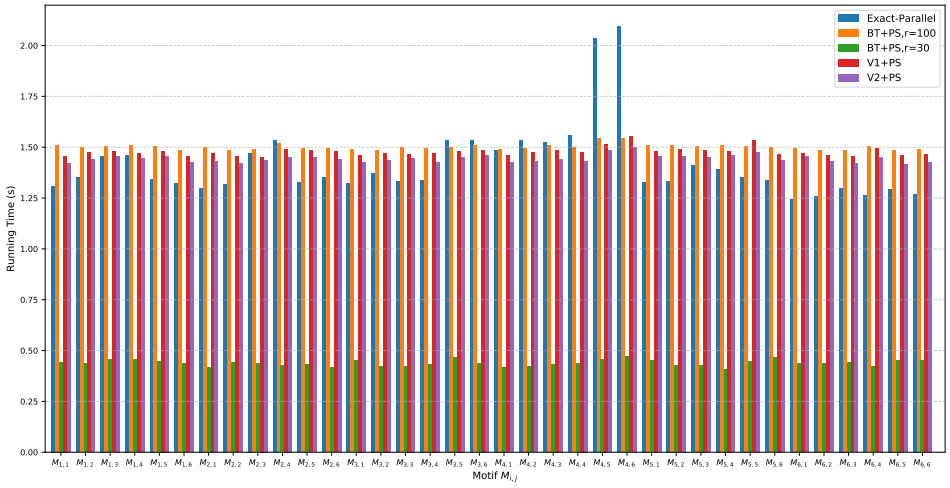


Figure 5.15: Running times for each motif on **AskUbuntu** dataset.

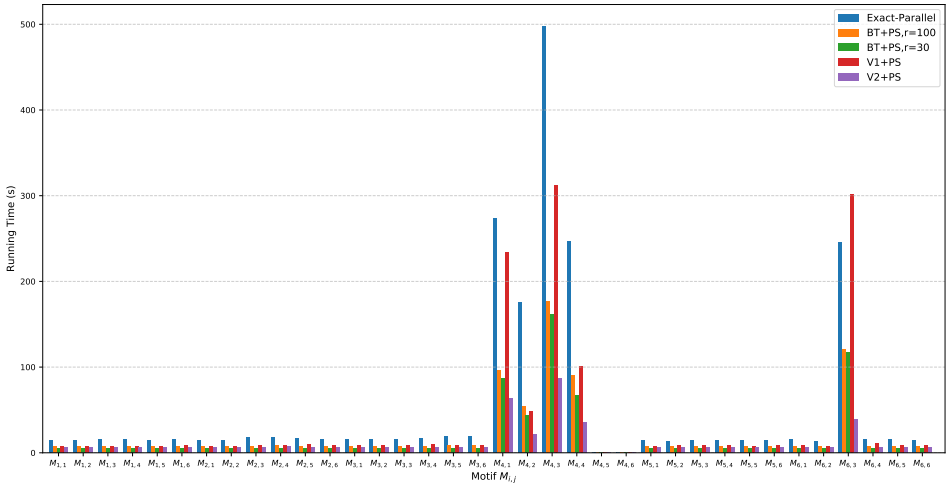


Figure 5.16: Running times for each motif on **Wikitalk** dataset.

Chapter 6

Conclusions

In this thesis we introduced, to the best of our knowledge, the first (ϵ, η)-approximation algorithms for the temporal motif counting problem, proved their guarantees, an efficiently computable bound on the sample size, and analyses of the variance of the estimate. We also provided additional bounds, that unfortunately are not computable, where we used the tool of Martingales.

Aside from the (ϵ, η)-approximation algorithms, we also developed an exact parallel routine which may become very efficient and scalable in practice. We implemented and compared all these techniques with the state of the art techniques in the field of mining motifs in temporal networks. Our results show that the exact parallel routine works very well in practice, especially on large datasets, where an exact routine cannot be adopted, thanks to it's scalability. Unfortunately for the (ϵ, η)-approximation algorithms it comes out that the sample size we derived it is too loose so it cannot be used in practice, moreover more samples than the state of the art heuristics are needed, to achieve equal or better performances. The estimate used in our algorithms has also a large variance, which makes it difficult to understand how the approximation works in practice. A positive note is that such techniques are fast and scalable and can handle large datasets.

In future we would like to investigate different techniques for improving the quality of our (ϵ, η)-approximation algorithms, in particular we want first to test such algorithms on larger datasets to see their approximation quality. As it comes out from the experiments such techniques seem to work better than the state of the art techniques when δ is small, we will investigate such direction. Then we would like to explore some variance reduction techniques for our estimate and understand if it is possible, thanks to such techniques, to concentrate the result of the algorithms around the desired estimate. We would also like to try to improve the bound on the sample size s, which may be done through a parametric analysis on the quantities involved in it's estimation.

Further future works may be to investigate different sampling techniques,

for example based on a fixed memory, such technique it would be very useful since as it comes out from our experiments the state of the art technique use a lot of memory; in fact some runs went out of memory on not so large datasets. Other directions for new sampling techniques could be to consider also the topology of the graph when constructing the sample, and not only the temporal dimension as we have done in the algorithms presented in this thesis.

Bibliography

[1] Aggarwal, Charu ; Subbian, Karthik: Evolutionary network analysis: A survey. In: *ACM Computing Surveys (CSUR)* 47 (2014), Nr. 1, S. 10

[2] Bello-Orgaz, Gema ; Jung, Jason J. ; Camacho, David: Social big data: Recent achievements and new challenges. In: *Information Fusion* 28 (2016), S. 45–59

[3] De Stefani, Lorenzo ; Terolli, Erisa ; Upfal, Eli: Tiered sampling: An efficient method for approximate counting sparse motifs in massive graph streams. In: *2017 IEEE International Conference on Big Data (Big Data)* IEEE, 2017, S. 776–786

[4] Holme, Petter: Modern temporal network theory: a colloquium. In: *The European Physical Journal B* 88 (2015), Nr. 9, S. 234

[5] Holme, Petter ; Saramäki, Jari: Temporal networks. In: *Physics reports* 519 (2012), Nr. 3, S. 97–125

[6] Jiang, Chuntao ; Coenen, Frans ; Zito, Michele: A survey of frequent subgraph mining algorithms. In: *The Knowledge Engineering Review* 28 (2013), Nr. 1, S. 75–105

[7] Kovanen, Lauri ; Karsai, Márton ; Kaski, Kimmo ; Kertész, János ; Saramäki, Jari: Temporal motifs in time-dependent networks. In: *Journal of Statistical Mechanics: Theory and Experiment* 2011 (2011), Nr. 11, S. P11005

[8] Leskovec, Jure ; Rajaraman, Anand ; Ullman, Jeffrey D.: *Mining of massive datasets*. Cambridge university press, 2014

[9] Liu, Paul ; Benson, Austin R. ; Charikar, Moses: Sampling Methods for Counting Temporal Motifs. In: *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining* ACM, 2019, S. 294–302

[10] Mackey, Patrick ; Porterfield, Katherine ; Fitzhenry, Erin ; Choudhury, Sutanay ; Chin, George: A chronological edge-driven approach to temporal subgraph isomorphism. In: *2018 IEEE International Conference on Big Data (Big Data)* IEEE, 2018, S. 3972–3979

[11] Mitzenmacher, Michael ; Upfal, Eli: *Probability and computing: Randomization and probabilistic techniques in algorithms and data analysis*. Cambridge university press, 2017

[12] Paranjape, Ashwin ; Benson, Austin R. ; Leskovec, Jure: Motifs in temporal networks. In: *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining* ACM, 2017, S. 601–610

[13] Pržulj, Nataša: Biological network comparison using graphlet degree distribution. In: *Bioinformatics* 23 (2007), Nr. 2, S. e177–e183

[14] Stefani, Lorenzo D. ; Epasto, Alessandro ; Riondato, Matteo ; Upfal, Eli: Triest: Counting local and global triangles in fully dynamic streams with fixed memory size. In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 11 (2017), Nr. 4, S. 43

[15] Sun, Xiaoli ; Tan, Yusong ; Wu, Qingbo ; Chen, Baozi ; Shen, Changxiang: TM-Miner: TFS-Based Algorithm for Mining Temporal Motifs in Large Temporal Network. In: *IEEE Access* 7 (2019), S. 49778–49789

[16] Tu, Kun ; Li, Jian ; Towsley, Don ; Braines, Dave ; Turner, Liam D.: Network classification in temporal networks using motifs. In: *arXiv preprint arXiv:1807.03733* (2018)

[17] Viard, Tiphaine ; Fournier-S'Niehotta, Raphaël ; Magnien, Clémence ; Latapy, Matthieu: Discovering patterns of interest in ip traffic using cliques in bipartite link streams. In: *International Workshop on Complex Networks* Springer, 2018, S. 233–241

[18] Wang, Jianxin ; Li, Min ; Wang, Huan ; Pan, Yi: Identification of essential proteins based on edge clustering coefficient. In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 9 (2011), Nr. 4, S. 1070–1080

[19] Yin, Hao ; Benson, Austin R. ; Leskovec, Jure: The local closure coefficient: a new perspective on network clustering. In: *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining* ACM, 2019, S. 303–311