Mining Microscopic and Macroscopic Changes in Network Data Streams

Corrado Loglisci[a,b,*], Michelangelo Ceci[a,b], Angelo Impedovo[a], Donato Malerba[a,b]

[a] *Department of Computer Science, Universita' degli Studi di Bari Aldo Moro, Bari, Italy*
[b] *CINI, Consorzio Interuniversitario Nazionale per lInformatica, Bari, Italy*

Abstract

Network data streams are unbounded sequences of complex data produced at high rate which represent complex systems that evolve continuously over time. In this scenario, a problem worthy of being studied is the analysis of the changes, which may concern a complex system as a whole or small parts of it. In this paper, these are distinguished into *macroscopic* changes and *microscopic* changes: macroscopic changes have impact on a substantial part of the network, whereas microscopic changes concern variations occurring in specific portions of the network. The algorithm we propose, called KARMA, combines the frequent pattern mining framework with an automatic time-window detection approach. In this way, it is able to detect changes on the frequent subnetworks mined from different time-windows: network changes are then represented as variations of structural regularities frequently observed over the stream. KARMA takes an holistic perspective, in which the two kinds of change are related each other. This is the main novelty with respect to the recent studies, which do not simultaneously extract microscopic and macroscopic changes. Experiments on several real-world network data streams show the effectiveness and efficiency of our approach in comparison with competing algorithms and the usefulness of the changes detected.

1. Introduction

Data streams are unbounded and time-ordered sequences of continuously incoming elements, which arrive at a rapid rate. One of the main characteristics of data streams is that they are often subject to frequent changes of the probability distribution. The analysis of these changes is attracting growing attention because it gives the opportunity to follow and understand the variations of the underlying process, adapt tools and services to new demands, as well as capture and delay undesirable alterations. Applications can be found in many domains, for example in telecommunication networks, where it is important to identify the variations in the use of the infrastructure and anticipate new demands. Other examples can be found in ecology, where it is important to identify pollution problems in time so that appropriate actions can be taken or in the context of social networks, where it can be useful to understand the variations of political and commercial orientation of the users.

Consequently, researchers are required to design techniques able to process data streams, detect a change and explain the nature of the change [1]. The task is not trivial for two main reasons. First, contrarily to other data stream mining tasks (e.g., anomaly detection), we cannot rely on a reference system, which recognizes rare and sporadic data, because the change can trigger a new data distribution and thus can be associated to manifestations of the underlying process. Second, the data produced in the recent streaming environments (e.g., social media, sensor technologies) are intrinsically complex because they refer to aggregations of multiple entities, which are heterogeneous in nature and can be inter-connected. So, most of existing works, originally designed for (simple) data streams (e.g., [2, 3]) become inapplicable.

A promising approach for handling complex data streams [4, 5] is the representation in form of *evolving network*, considered the natural peculiarity to account for entities of different nature (nodes), which can be related each other (edges). This solution allows us to represent macroscopic and microscopic changes of the complex data stream, that

*Corresponding author
Email address: corrado.loglisci@uniba.it (Corrado Loglisci)

is, variations that regard the complex data as whole and variations that regard only portions. Indeed, the macroscopic changes can be reflected as changes at the whole structure of the network, while those microscopic correspond to evolutions of subnetworks, such as changes in the properties of nodes and edges as well as the insertion and deletion of single nodes and edges.

To give a general idea of the change in network data streams, we report a concrete example in the domain of communication networks, where data streams are produced on the telecommunication transactions (e.g., call data records) of the users (an illustration is reported in Figure 1). The nodes of the network model users, while an edge reports the modality of communication (e.g., short messaging, voice call) between two users at a certain time in a geographic area. In this scenario, the changes can be due to variations of the attitude of the users to communicate or to failures of the service provider. This means that an edge, which connects two users in a time-point, can disappear in another time-point or can be replaced by another edge representing a different modality of communication. For instance, some users, which usually exchange short messages, start to communicate through instant messaging because the service of short messaging experiences failures. When associated to a few nodes or single subnetworks, these variations can concern small portions of the geographic area and thus denote microscopic changes, while when they are associated to the most part of the users, the changes are macroscopic because have impact on a substantial part of the users within the area.

It is evident that what happens at macroscopic level can be ascribed to a combination of evolutions occurring on single subnetworks and, conversely, changes occurring on the subnetworks can trigger evolutions of the whole structure of the network. Clearly, not all the microscopic changes can explain those macroscopic. Indeed, changes which concern few subnetworks and which happen sporadically could be not interesting compared to those which occur many times on the same subnetwork or those which involve many subnetworks. To capture this behavior, it could be helpful to consider the notion of frequency and determine the change of the subnetworks on the basis of the variations of their frequency.

To identify these variations, we may act directly at the level of the network data, but this is costly because it requires the analysis of single nodes and edges over the entire stream. Instead, we propose the use of frequent patterns, which provide an abstract form of the data and allow us to work jointly on microscopic and macroscopic changes. So, single patterns summarize specific subnetworks (subgraphs) and may be associated to microscopic changes, while sets of patterns synthesize different subnetworks (even the whole network) and may be associated to macroscopic changes.

In this paper, to detect macroscopic and microscopic changes from network data stream, we propose a method which combines the time-window model based on *landmark* windows [6] with frequent subnetworks. The landmark windows have increasing size and, in this work, are used to collect the data that come one by one in time until a landmark, that is, a macroscopic change, is detected. So, the method mines frequent subnetworks from the landmark windows by keeping subnetworks and their frequencies updated. In these terms, macroscopic changes correspond to the updates on the set of frequent subnetworks, while microscopic changes correspond to the updates of the frequency of the subnetworks. To quantify the magnitude of the change we use quantitative measures for both kinds of changes. It is should be noted that this model of analysis complies with the one used to process online data streams and differs from the analysis of the offline data streams in which the data are collected by windows of regular fixed size and processed after storage, as in a data warehouse system [7, 8].

As an example, consider the telecommunication transactions of four users recorded in two hours and suppose the landmark window covers the time-interval [7:00,7:30] in its initial size (Figure 1). We observe that the status of the communications generally changes over the considered time-interval [7:00,8:15]. This would denote a macroscopic change in the modalities of communication of the four users and thus a significant variation in the use or in the offer of the services of a provider on the geographic area. A closer analysis reveals the presence of different structural variations concerning several portions of the whole network as the landmark window collects newly incoming transactions. One of the more evident variations is the one related to the occurrences of “subnetwork 1”, it is always present in the window [7:00,7:30] (out of the three transactions of [7:00,7:30]), while, it disappears in the next three transactions (out of the six transactions of [7:00,8:15]). So, the corresponding microscopic change lies in the change of the frequency, which is very high when observed in the window [7:00,7:30], while it decreases when observed in the window [7:00,8:15]. Another change is associated with “subnetwork 2”, which is not present in the window [7:00,7:30], while it becomes frequent as the window collects the next three transactions. No contribution to the macroscopic change can be attributed to “subnetwork 3”, which has only one occurrence out of the three transactions of [7:00,7:30] and continues to have one occurrence in the next three transactions. Besides the numeric variation, we can interpret these

behaviors with respect to their impact on the telecommunication domain. For instance, "subnetwork 1" may reveal that the three users ("user B", "user C", "user D") switch from "short messaging" to "instant messaging", and this could trigger other changes, for instance, the use of "short messaging" by the users "user A" and "user B". Obviously, the size of landmark windows determines the granularity of the changes and, in our approach, many landmark windows with automatically determined sizes are considered. This means that in the domain of telecommunication transactions, changes can refer to few hours (as in the example reported in Figure 1) or to days, weeks etc.

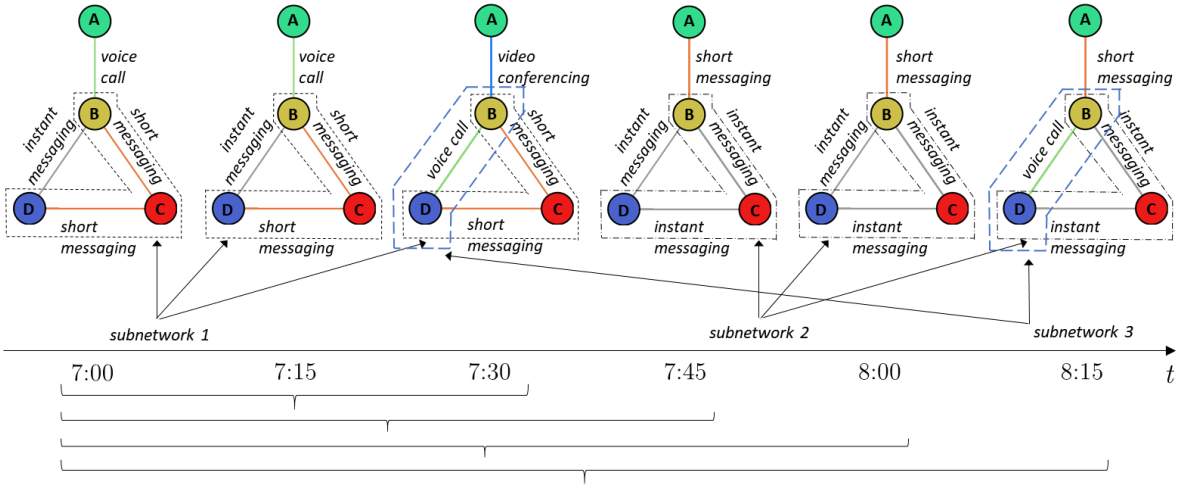


Figure 1: Representation of the scenario of the telecommunication in the form of evolving network.

The rest of this paper is structured as follows. We introduce and discuss related works in Section 2. Then, the problem faced in the paper is formally stated in Section 3. In Section 4, the proposed computational solution, called KARMA (networK streAm macRoscopic Microscopic chAnges), is described. We structure it in four main steps and describe the algorithmic details. The experimental setting is detailed in Section 5, where the results are also reported and discussed. Finally, in Section 6, conclusions are drawn and future research directions are identified.

2. Related Work and Contribution

In this section we discuss the recent literature under three different perspectives and emphasize the novelty of the current paper for each perspective.

The first perspective concerns the problem of change detection. A typical assumption, also considered in this work, is the absence of any ground truth able to establish which data represent a change and which do not. A different solution would be based on labeling, but this requires manual intervention, is subjective and is time-consuming due to the complexity of the network and to the size of the stream. For these reasons, for change detection, unsupervised techniques are more attractive and viable than supervised ones.

In unsupervised change detection we can identify two main research lines, the first one focuses on the discovery of variations concerning the network as a whole and thus investigates what we name macroscopic changes in Section 1, while the second line is addressed to the characterization of variations local to portions of the network and thus considers what we name microscopic changes in Section 1. Then, we may ascertain that the current studies cover only some aspects of the problem at hand and that they might be limited to face only sub-problems. As our best knowledge, there are very few attempts able to provide a holistic way to the problem while detecting both the variations of the whole network and the variations on local portions [9]. Simultaneously extracting macroscopic and microscopic changes in the practical scenario of the streaming network data represents the main novelty of the current paper.

One of the major challenges when dealing with data streams, which are typically generated in non-stationarity environments and characterized by a dynamically changing data distribution, is the phenomenon of the *concept drift*.

Widmer and Kubat [10] introduced the notion of concept drift by distinguishing the *real* concept drift from *virtual* concept drift: the former reflects real changes occurring in the environments, the latter does not occur really in the environment but, rather, in the description of the environment done with the data streams, which closely concerns the data distribution. Delany et al. [11] and Sun et al. [12] revised the notion of virtual concept drift by distinguishing those *sudden* (abrupt and instantaneous) from those *gradual* (moderate and slow). We fully consider this notion since our method uses a flexible windowing mechanism to catch both sudden and gradual changes.

The second perspective concerns the methodological approach. We resort to the frequent pattern mining framework as solution to generate a summarizing form of the network data stream. We should note that frequent patterns (subnetworks) are not the primary objective of the paper, but the means to capture two kinds of changes. There are several works in the literature that use a similar approach: in [13] the authors study the problem of analyzing the whole evolution of the network and propose a method which creates a graph (i.e. a set of patterns), where conserved states of the network are the vertices, while the admissible transitions among those states are the edges. The conserved states correspond to sequences of consecutive time-stamped networks with structural similarity. They are represented as induced subgraphs whose configuration of the relations (labelled edges) and nodes occurs frequently over the corresponding sequence. The transitions are associated to modifications on the nodes of a state and can determine the migration towards the next state. The paths of the graph of the states represent alternative courses which can characterize the whole evolution. Only those which are considered maximal are retained. Berlingerio et al. [14] proposed to extract "Graph evolution rules", that is, graphs where nodes denote entities and edges are labeled with the time-stamp. In this way it is possible to model the evolution of the entities. Graph evolution rules are extracted from a sequence of snapshots of an evolving graph by resorting to frequent pattern mining solutions. Although the problem solved in this work is similar to ours, this approach is only able to represent insertions and not deletions of nodes/edges. The notion of "Co-evolution pattern" [15] has been introduced to model simultaneously occurring evolutions. It aims at capturing the change of attributed dynamic graphs at different levels of granularity. An attributed graph is a particular kind of graph with nodes described by a set of attributes. Thus, a co-evolution pattern represents a subgraph whose attribute values follow the same evolution. Each subgraph is associated with two kinds of information, that is, the trend of the attribute values and time-points in which the attribute has that trend. Compared to our work, there are several differences, the main one is that does not work in the streaming scenario.

Changes in data streams have been recently considered in the algorithm CD-TDS [16], which, similarly to our, explores the problem at two levels of granularity by defining two different types of change, "local changes" and "global changes". The former refer to changes in the distribution of the data and correspond to quantitative variations, that is, changes of the frequency of individual items. The latter refer to changes in the connections among the data and correspond to structural variations, that is, changes of co-occurrences of items without taking variations of the frequency into account. Thus, they denote evolutions of different nature, but, contrary to our work, they are not related each other. In fact, local changes are detected independently from the presence of global changes. In [17] we investigated the task of characterizing the dynamics by introducing the notion of evolution chains. In that method frequent patterns represent subgraphs which exhibit frequent label changes. These changes are discovered over consecutive and predefined time-periods and then combined in sequence to model the evolution of the network. More recently, in [18] we have studied the same problem within a logic programming framework, which enables the integration of background knowledge on the network and the representation of temporal information associated to the changes. In particular, we studied the task of capturing variations exhibited by the patterns discovered from a data network over time. Although these two methods capture the whole evolution of the network, they work on local portions of the network by disregarding changes which can concern its whole structure. Also, they assume that the changes occur necessarily in pre-defined time-periods, which makes their application rather limited in the case of streaming environments. Nohuddin et al. [19] track changes in the trend of statistical parameters of frequent patterns in social networks. The patterns are mined from descriptive properties of the nodes of the networks and therefore do not provide information on the structural changes of the network.

Since the aforementioned methods consider evolving networks data which are not necessarily associated to data streams, the third perspective concerns change detection in the streaming setting. In this field, extensive research has been done on the problem of anomaly (or outlier) detection, where the task is to identify items, events or observations of the data stream which do not conform to the expected behavior. Anomalies occur infrequently and appear as abrupt changes [20]. One of the most recent works is reported in [21]. It extends the traditional spectral-based method to address the issue of the scalability of the network stream. In particular, the authors combine the compress sensing

theory with random projection, to handle large-scale networks as compressed data. Normal behavior is modeled with the principal subspace, while the anomalies are captured in the residual subspaces.

Another research line focuses on the identification of correlated evolutions. For instance, [22] describe a graph-theory inspired approach to discover changing components. They report an algorithm to efficiently mine frequent components by combining the density of the evolutions and the size of the changed subgraphs. A problem which is more similar to ours has been investigated in [23], where the aim is to identify "burst areas", that is, temporal windows that show significant differences with respect to previous windows (by looking at the snapshots in each window). In this work, the authors propose working on the counts of the changes that the weights of edges exhibit. The technique uses a wavelet tree to maintain the counts of the changes. This gives the opportunity to incrementally identify burst areas of different sizes without necessarily working on the complete representation of the periods involved. The main difference with respect to our approach is that [23] do not resort to the concept of *frequent* subnetworks but propose to define bursts as subgraphs by exploiting the changes of the weights associated to the edges. Finally, the use of the frequency in the change detection problem has been investigated also for unstructured data. Zhang et al. [24] propose "scientific evolutionary pathways" to track evolutions of scientific research areas in terms of changes of the composition of involved terminology and changes of the frequency of involved key-words.

3. Background and problem definition

Before formally stating the data mining problem, we provide the notations for data representation and basic definitions for the rest of the paper. Then, we describe the concepts of macroscopic and microscopic changes by reporting the necessary notions. Finally, we provide a formal definition of the problem to be solved.

3.1. Data representation

Let $D = \langle D_1, D_2, \ldots, D_n, \ldots \rangle$ be the time-ordered stream of networks. At each time-point τ_i, the stream has a snapshot of the network $D_i = (N_i, E_i)$, where N_i and E_i denote the sets of nodes and edges observed in τ_i, respectively. For our convenience, we also define the complete set of nodes $N = \bigcup_i N_i$ and the complete set of edges $E = \bigcup_i E_i$. Since the definition of N and definition of E assume that we know the complete sets of nodes and edges, we consider them to be "active" sets, that is, sets that are continuously updated and that are defined according to the information we already have. Intuitively, we can imagine that the stream D records the state of a network defined on the sets (N, E) and produces an unbounded sequence of time-stamped snapshots of such a network.

We assume that each node and edge is labeled and that two nodes can be linked by multiple labeled edges with different labels. Labels from the nodes are taken from the set $\mathcal{N}$, while labels for the edges are taken from the set $\mathcal{E}$. Therefore each snapshot D_i is a labeled multi-graph extensively described by the set $D_i = \{(u, v, e) \mid u, v \in N_i, e \in \mathcal{E}\}$. An element (u, v, e) is termed as *triple*. A triple occurs at the time-point τ_i if the snapshot of the network D_i includes it.

A *landmark window* (or, simply, *window*) $W = [\tau_i, \tau_j]$ is the sequence of consecutive time-points $\{\tau_i, \ldots, \tau_j\}$. Following [25], a (landmark) window W'' is successive to another (landmark) window W' when they share some initial time-points, that is, $W' = [\tau_i, \tau_n]$ and $W'' = [\tau_i, \tau_m]$, with $\tau_i < \tau_n < \tau_m$. For simplicity, we use W to refer also the sequence of snapshots $\{D_i, \ldots, D_j\}$ observed at the time-points $\{\tau_i, \ldots, \tau_j\}$ respectively. Consequently, the width $|W|$, corresponding to the number of time-points in W ($|W| = j - i + 1$), is equal to the number of the snapshots collected in W.

In the following we report an example of time-ordered stream in the context of the telecommunication network. The example shows a time-ordered stream in which each snapshot registers the state of the network every 15 minutes (Figure 2 and Table 1). Also, the example shows two windows $W' = [7{:}00, 8{:}00]$ and $W'' = [7{:}00, 8{:}45]$ that collect five and eight snapshots respectively.

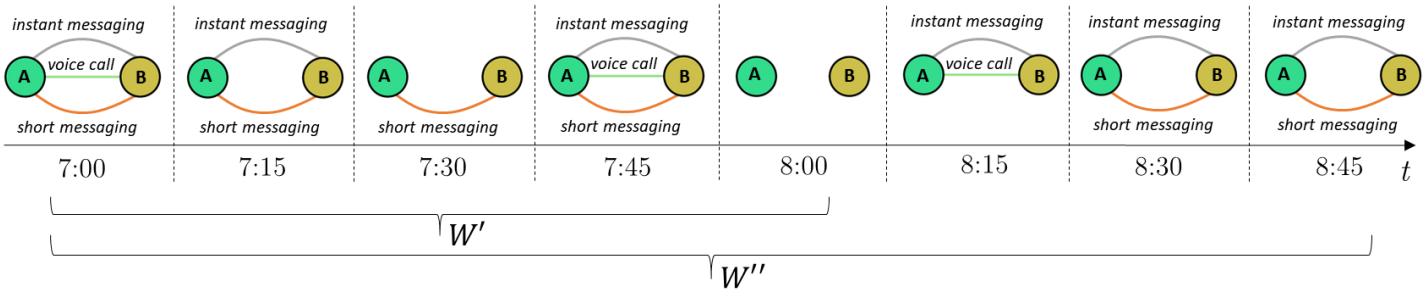


Figure 2: A time-ordered stream in the context of the telecommunication network. Each snapshot registers the state of the network every 15 minutes.

Table 1: The time-ordered stream represented as sets of triples associated to their respective time points.

τ_i	D_i
7:00	{(user_A, user_B, voice_call),(user_A, user_B, instant_messaging), (user_A, user_B, short_messaging)}
7:15	{(user_A, user_B, instant_messaging),(user_A, user_B, short_messaging)}
7:30	{(user_A, user_B, short_messaging)}
7:45	{(user_A, user_B, voice_call),(user_A, user_B, instant_messaging), (user_A, user_B, short_messaging)}
8:00	$\emptyset$
8:15	{(user_A, user_B, voice_call),(user_A, user_B, instant_messaging)}
8:30	{(user_A, user_B, instant_messaging),(user_A, user_B, short_messaging)}
8:45	{(user_A, user_B, instant_messaging),(user_A, user_B, short_messaging)}
...	...

W' W''

3.2. Basics

Let $P = \{(u, v, e) \mid u, v \in N, e \in \mathcal{E}\}$ be a pattern expressed as a set of triples, we denote as $|P|$ the *length* of the pattern P. The *support* $sup_W(P)$ of a pattern P is its relative frequency in a window W, which is computed as the fraction of the network snapshots collected in W in which P occurs.

$$sup_W(P) = \frac{|\{D_i \in W \mid P \subseteq D_i\}|}{|W|} \tag{1}$$

This means that we simplify the mechanism of counting of the occurrences of a pattern P in a snapshot D_i by checking the presence/absence of its triples. The *support* has values in [0,1].

Let W be a window and P be a pattern. If the support $sup_W(P)$ in W is greater than or equal to $minSUP$, then P is *frequent* in W. This means that P occurs in at least a set of size $minSUP$ of the snapshots collected in W ($minSUP \in [0, 1]$ is a user-defined threshold). Conversely, if the support $sup_W(P)$ is less than $minSUP$, then P is *infrequent*

Definition 1 (Subnetwork). *Let P be a pattern and $N_P = \{v \in N \mid \exists(u, v, e) \in P \vee (v, u, e) \in P\})$ be the set of nodes involved in P. Then, P is a subnetwork if, for each pair of nodes $(a, b) \in N_P \times N_P$, there exists a path in P that connects a and b.*

Intuitively, the subnetworks correspond to patterns in which it is possible to reach every node from all the other nodes, therefore they are connected subgraphs. However, in the rest of the paper we use the term subnetworks to specifically refer to subgraphs represented in form of sets of triples. For brevity, we denote subnetworks which are frequent patterns as *frequent subnetworks*.

3.3. Macroscopic changes

As introduced in Section 1, macroscopic changes correspond to changes on the set of the frequent subnetworks and on the set of the infrequent subnetworks. To handle these sets and efficiently detect the changes, we use a lattice structure, which is largely used in problems of frequent itemsets mining [26].

A lattice $\mathcal{L}_{\geqslant}$ is a partially-ordered set and, in this work, it is defined according to a generality order $\geqslant$ based on the binary relation of subset-containment between subnetworks. The lattice is organized by levels of generality: given P and Q two subnetworks, we say that P is more general than Q ($P \geqslant Q$), if $P \subseteq Q$ (for instance, P_1 and P_4 in Figure 3). More precisely, at the same level of $\mathcal{L}_{\geqslant}$ we have subnetworks of the same length but that are not related by the generality order $\geqslant$. At the (k+1)-th level there are subnetworks with (k+1) triples which are one more with respect to the triples of the subnetworks at the k-th level.

The generality order $\geqslant$ is monotonic with respect to the support, thus whenever P_1 is infrequent, its more specific subnetworks (e.g., P_4 and P_5) will be infrequent too.

The structure of the lattice depends on N and $\mathcal{E}$ and does change over time; what changes is the frequency associated to the subnetworks, as a consequence of the arrival of new snapshots. This results in the update of the set of the frequent subnetworks and of the set of the infrequent subnetworks: subnetworks which were frequent (infrequent) can become infrequent (frequent). Therefore, we search for macroscopic changes as variations of the set of the frequent and of the set of the infrequent subnetworks on two successive landmark windows. To quantify these variations, we use the following formula (*macroChange*):

$$MC(\mathcal{L}', \mathcal{L}'') = \frac{|\mathcal{F}_{\mathcal{L}''} - \mathcal{F}_{\mathcal{L}'}| + |\mathcal{F}_{\mathcal{L}'} - \mathcal{F}_{\mathcal{L}''}|}{|\mathcal{F}_{\mathcal{L}'}| + |\mathcal{F}_{\mathcal{L}''} - \mathcal{F}_{\mathcal{L}'}|}, \tag{2}$$

where, given two successive windows W' and W'' and their associated lattices $\mathcal{L}'$ and $\mathcal{L}''$, $\mathcal{F}_{\mathcal{L}'}$ and $\mathcal{F}_{\mathcal{L}''}$ are the sets of frequent subnetworks identified on $\mathcal{L}'$ and $\mathcal{L}''$ respectively. The lattices $\mathcal{L}'$ and $\mathcal{L}''$ have the same structure, but differ on the support values that are computed on the snapshots W' and on the snapshots W'' respectively. The set $(\mathcal{F}_{\mathcal{L}''} - \mathcal{F}_{\mathcal{L}'})$ comprises the subnetworks which have become frequent in W'', while $(\mathcal{F}_{\mathcal{L}'} - \mathcal{F}_{\mathcal{L}''})$ is the set of subnetworks which have became infrequent in W''. Intuitively, the macroscopic change is expressed by the differences between the sets $\mathcal{F}_{\mathcal{L}'}$ and $\mathcal{F}_{\mathcal{L}''}$. The *macroChange* has values in [0,1].

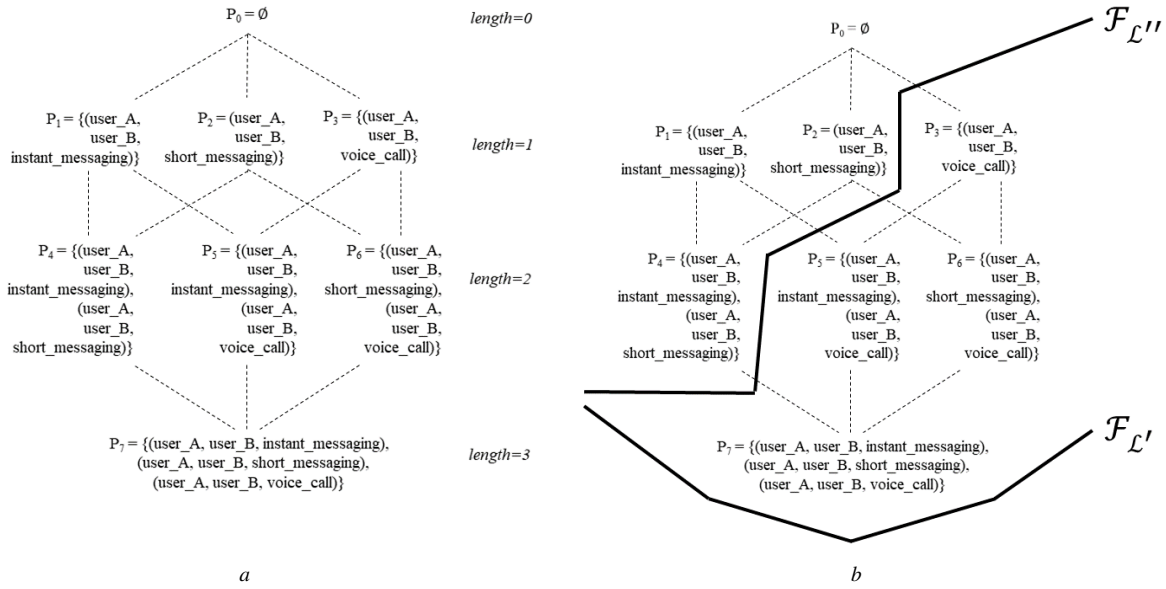


Figure 3: a) Representation of the lattice $\mathcal{L}$ of subnetworks arranged by generality order $\geqslant$. b) Representation of the macroscopic change between W' and W''. The two lines in bold denote the set of frequent subnetworks in W' ($\mathcal{F}_{\mathcal{L}'} = \{P_0, P_1, P_2, P_3, P_4, P_5, P_6, P_7\}$) and the set of frequent subnetwork in W'' ($\mathcal{F}_{\mathcal{L}''} = \{P_0, P_1, P_2, P_4\}$). The value of *macroChange* is computed upon the two differences between the sets $\mathcal{F}_{\mathcal{L}'}$ and $\mathcal{F}_{\mathcal{L}''}$ ($\mathcal{F}_{\mathcal{L}'} - \mathcal{F}_{\mathcal{L}''} = \{P_3, P_5, P_6, P_7\}$ and $\mathcal{F}_{\mathcal{L}''} - \mathcal{F}_{\mathcal{L}'} = \emptyset$ respectively).

As an example, consider the lattice of subnetworks reported in Figure 3 and the snapshots shown in Table 1. Suppose that their support values in $\mathcal{L}'$ (built on $W' = [7{:}00, 8{:}00]$) are the following: $sup_{W'}(P_1) = 0.6$, $sup_{W'}(P_2) = 0.8$, $sup_{W'}(P_3) = 0.4$, $sup_{W'}(P_4) = 0.6$, $sup_{W'}(P_5) = 0.4$, $sup_{W'}(P_6) = 0.4$, $sup_{W'}(P_7) = 0.4$. Whereas in $\mathcal{L}''$ (built on $W'' = [7{:}00, 8{:}45]$) the support values are $sup_{W''}(P_1) = 0.75$, $sup_{W''}(P_2) = 0.75$, $sup_{W''}(P_3) = 0.25$, $sup_{W''}(P_4) = 0.416$, $sup_{W''}(P_5) = 0.25$, $sup_{W''}(P_6) = 0.166$, $sup_{W''}(P_7) = 0.166$.

By assuming the threshold $minSUP$=0.35, we observe that all the subnetworks belong to the set of the frequent subnetworks $\mathcal{F}_{\mathcal{L}'}$ when considering the snapshots of the window W'. Instead, when considering the snapshots of the window W'', the subnetworks P_1, P_2 P_4 remain in the set of the frequent subnetworks $\mathcal{F}_{\mathcal{L}''}$, while the subnetworks P_3, P_5, P_6, P_7 become infrequent, so the value of *macroChange* is 0.57, as shown in Figure 3*b*.

3.4. Microscopic changes

In this work, we suppose that macroscopic changes are triggered by a combination of microscopic changes, which we formulate by adapting the concept of emerging patterns [27]. In their classical use, the emerging patterns work on static data to solve a classification task through a discriminative approach. The patterns whose frequency greatly differs from classes are considered to be emerging. We use them in a different scenario (that is, evolving data) and for a new purpose (that is, characterizing the evolution of the network over a stream of snapshots). More precisely, the emerging patterns model subnetworks whose occurrences change between two successive landmark windows.

Definition 2 (Emerging Subnetworks (ES)). *Let W' and W'' be two successive landmark windows, $sup_{W'}(P)$ and $sup_{W''}(P)$ be the support of a subnetwork P in W' and in W'', respectively. P is emerging if*

$$\frac{sup_{W'}(P)}{sup_{W''}(P)} > minGR \quad \vee \quad \frac{sup_{W''}(P)}{sup_{W'}(P)} > minGR, \tag{3}$$

where $minGR(\geq 1)$ is a user-defined minimum threshold.

The formulas reported above allow us to quantify the magnitude of the changes for each subnetwork, that is, at the microscopic level. While the ratio $\frac{sup_{W''}(P)}{sup_{W'}(P)}$ (denoted as *growth-rate*, $GR_{W',W''}(P)$) models the increase of the support

from W' to W'', the ratio $\frac{sup_{W'}(P)}{sup_{W''}(P)}$ ($GR_{W'',W'}(P)$) models the decrease of the support from W' to W''. According to Dong and Li [27], we assume that $GR(P) = \frac{0}{0} = 0$ and $GR(P) = \frac{>0}{0} = +\infty$.

An example of emerging subnetwork in the context of the telecommunication network can be drawn over the windows W' = [7:00, 8:00] and W'' = [7:00, 8:45]:

P_6 : {($user\_A, user\_B, short\_messaging$), ($user\_A, user\_B, voice\_call$)}
where [$sup_{W'}(P_6) = 0.4$] and [$sup_{W''}(P_6) = 0.166$].

Here, the value of the growth-rate $GR_{W'',W'}(P_6)$ is 2.4 (0.4/0.166). If *minGR*=1.5, the subnetwork P_6 is considered emerging and it expresses a change in the support of a portion of the whole network. More precisely, it denotes the same subnetwork (see Definition 1, Figure 3*b*) which has a decreasing number of occurrences from W' to W''.

3.5. Formal definition of the problem

We can now give a formal statement of the problem of mining macroscopic and microscopic changes:
Given:

- A stream of snapshots of the network $D = \langle D_1, D_2, \ldots, D_n, \ldots \rangle$
- A threshold $minSUP \in [0, 1]$, which represents the minimum support value for mining frequent subnetworks
- A threshold $minMC \in [0, 1]$, which represents the minimum value for detecting macroscopic
- A threshold $minGR \in [1, +\infty)$, which represents the minimum growth-rate for detecting microscopic changes changes

Find:

- The pairs of successive landmark windows (W', W'') for which $MC(\mathcal{L}', \mathcal{L}'') > minMC$, where
  - $\mathcal{L}'$ and $\mathcal{L}''$ are the lattices generated from W' and W'' (resp.), according to $minSUP$,
- For each pair of windows (W', W''), identified at the previous point, all the emerging subnetworks (according to *minGR*) which represent subnetworks.

All the three parameters $minSUP$, *minMC* and *minGR*, operate on the support of the subnetworks. As in many frequent pattern mining algorithms, the identification of the best values for input parameters is subjective and depends on the desired trade-off between completeness of the extracted knowledge and efficiency of the data mining process. In the experimental section we will investigate the effect of these parameters on the identified macroscopic and microscopic changes.

4. The Algorithm

For the problem formalized in the previous section, we propose the algorithm KARMA (Algorithm 1), which operates by orchestrating the execution of four main procedures.

1. Procedure 1 (Algorithm 1, lines 5-20) uses two time-window models (landmark window and sliding window) to handle incoming network snapshots and searches for changes on two successive landmark windows.
2. Procedure 2 (Algorithm 1, lines 1-4) initializes the lattice of the subnetworks when the stream analysis starts (see Figure 4*a*). Algorithmically, the lattice is initialized by means of a frequent subgraph discovery algorithm and, upon the extensions of the landmark windows, the lattice is updated and not reconstructed from scratch.
3. Procedure 3 (Algorithm 1, lines 6-9) concatenates the block Π of newly incoming network snapshots to the window W' and updates the support values of the subnetworks. Consequently, it updates the set of the frequent subnetworks and set of infrequent subnetworks (see Figure 4*a*, a block Π of network snapshots is concatenated to the window W').
4. Procedure 4 (Algorithm 1, lines 10-19) is in charge of detecting macroscopic changes. This is done by matching the lattice $\mathcal{L}'$, which has been updated with the snapshots of the window W', against the lattice $\mathcal{L}''$, which has been updated with the snapshots of the window $W'' = W' \cup \Pi$ (see Figure 4*b*). If there are not macroscopic changes (lines 16-19), KARMA considers the window W'' as W' (W''=$W' \cup \Pi$) and continues to process the stream D by taking a block Π' of new snapshots (see Figure 4*c*). On the contrary, the set of frequent subnetworks and the set of infrequent subnetworks change significantly. In this case, KARMA analyzes individually the subnetworks, in order to detect changes in terms of the support values (microscopic changes, lines 11-16). Once the analysis is completed, KARMA delivers the results relative to the (current) landmark windows W' and W'' and restarts by initializing the landmark window W' with the network snapshots that have caused the macroscopic changes, that is, the old block Π. The new window W'' will be created by concatenating a new block Π' of network snapshots to the window W' (see Figure 4*d*).

In the following subsections we provide further details on procedure *2*, *3* and *4*.

Algorithm 1: KARMA

inputs : $D, minSUP, minMC, minGR$

output: MCs `/* the set of macroscopic changes described in terms of time-windows and emerging subnetworks */`

1 $W'' \leftarrow \emptyset, \mathcal{L}' \leftarrow \emptyset, \mathcal{L}'' \leftarrow \emptyset$
2 $MCs \leftarrow \emptyset$
3 $W' \leftarrow initializeWindow(D)$
4 $\mathcal{L}' \leftarrow initializeLattice(W')$
5 **while** *a new block* Π *comes from* D **do**
6 $\quad W'' \leftarrow W' \cup \Pi$
7 $\quad \mathcal{L}'' \leftarrow updateSupport(\mathcal{L}', W'')$
8 $\quad \mathcal{F}_{\mathcal{L}'} \leftarrow \{P \in \mathcal{L}' \mid sup_{W'}(P) \geq minSUP\}$
9 $\quad \mathcal{F}_{\mathcal{L}''} \leftarrow \{P \in \mathcal{L}'' \mid sup_{W''}(P) \geq minSUP\}$
10 $\quad MC \leftarrow \frac{|\mathcal{F}_{\mathcal{L}''} - \mathcal{F}_{\mathcal{L}'}| + |\mathcal{F}_{\mathcal{L}'} - \mathcal{F}_{\mathcal{L}''}|}{|\mathcal{F}_{\mathcal{L}'}| + |\mathcal{F}_{\mathcal{L}''} - \mathcal{F}_{\mathcal{L}'}|}$
11 $\quad$ **if** $MC \geq minMC$ **then**
12 $\quad\quad ES \leftarrow computeEmergingSubnetworks(\mathcal{F}_{\mathcal{L}'}, \mathcal{F}_{\mathcal{L}''}, minGR)$
13 $\quad\quad MCs \leftarrow MCs \cup \{(\langle W', W'' \rangle, ES)\}$
14 $\quad\quad W' \leftarrow \Pi$
15 $\quad\quad \mathcal{L}' \leftarrow updateSupport(\mathcal{L}', \Pi)$
16 $\quad$ **else**
17 $\quad\quad W' \leftarrow W''$
18 $\quad\quad \mathcal{L}' \leftarrow \mathcal{L}''$
19 $\quad$ **end**
20 **end**

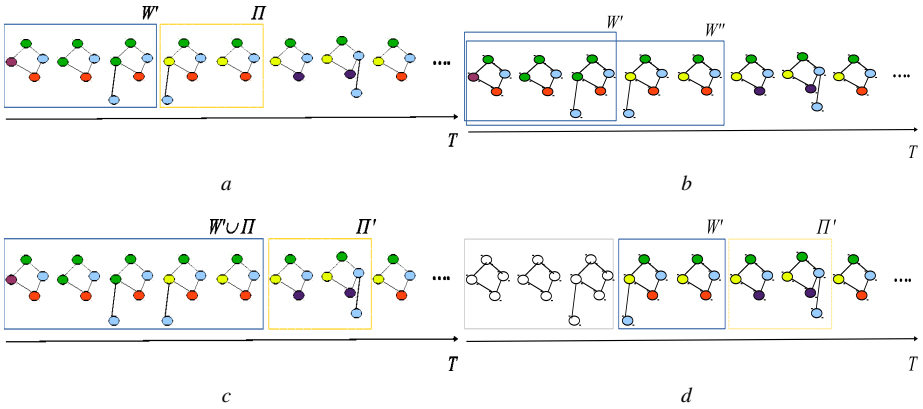


Figure 4: The network stream is analyzed through landmark window models: given an initial time window W' and the data block Π (a), two successive windows W' and $W'' = W' \cup \Pi$ are considered (b). If no change is detected between W' and W'', the extension of the window W' with Π is kept (c), otherwise the analysis re-starts with a new landmark window, that is, the window W' shifted forward (d).

4.1. Construction of the lattice and discovery of frequent subnetworks

To build the lattice there are two options. First, using existing algorithms of frequent subgraph mining. Second, adapting existing algorithms of frequent pattern mining originally designed for streaming data. We follow the second alternative since many solutions of frequent subgraph mining proposed in the literature *i)* are not designed to work on streaming scenario, *ii)* could not handle multi-edges networks, that is, network allowing more than one edge among nodes and *iii)* consider only frequent subnetworks, while we need to track also those infrequent. Our solution revises the algorithm proposed in [28] and extends it to *i)* handle subgraphs represented as sets of triples (this is necessary in order to model the network), *ii)* model streaming data as stream of (sets of) triples, instead of working on classical itemsets, *iii)* work on streaming scenario through an efficient computation of the support values without accessing the lattice, *iv)* track the frequent subnetworks that at a certain time-point become infrequent and the infrequent subnetworks that at a certain time-point become frequent. An example of lattice built by KARMA is illustrated in Figure 5.

To implement the lattice formalized in Section 3.2, we use a set enumeration tree (SE-tree) [29], which is a data structure that arranges a search space according to a pre-defined order [30]. In this work, we use it to arrange the lattice over a tree, whose vertices are associated to the elements of the power set of the set of all the triples $\mathcal{P}(N \times N \times \mathcal{E})$ (see Figure 5). The vertices of different levels of the SE-tree are sorted by the generality order (solid arrows in Figure 5b), as formalized in Section 3.3. The order of the vertices of the same level relies on a preference criteria on triples that uses the lexicographic order on the labels of nodes and edges. More precisely, the vertices (subnetworks) of each level of the SE-tree are sorted as follows: given two triples (u_i, v_i, e_i) and (u_j, v_j, e_j), we first consider the lexicographic order between the labels of the nodes u_i and u_j, then the lexicographic order between the labels of the nodes v_i and v_j and finally the lexicographic order between the labels of the edges e_i and e_j.

For instance, the subnetworks

$$P_4 : \{(user\_A, user\_B, instant\_messaging), (user\_A, user\_B, short\_messaging)\}$$
$$P_5 : \{(user\_A, user\_B, instant\_messaging), (user\_A, user\_B, voice\_call)\}$$

are sorted in the order "first P_4 then P_5" (see Figure 5b) because the label *short_messaging* of the second triple of P_4 lexicographically precedes the label *voice_call* of P_5.

The procedure of generation of the subnetworks relies on a recursive strategy which works on the vertex of the k-th level to determine the vertices of the $(k+1)$-th level. In particular, it *1)* follows a depth-first search method and *2)* builds a subnetwork with $(k+1)$ triples, by combining two subnetworks with k triples that meet the established order (dashed arrows in Figure 5b). To generate a vertex of the level 2, we require only that the two vertices of the level 1 meet the lexicographic order. To generate a vertex of the level $(k+1)$, successive to the level 2, we require that *i)* the two vertices of the level k have the first $k-1$ triples in common and *ii)* the k-th triples of the two vertices meet the lexicographic order.

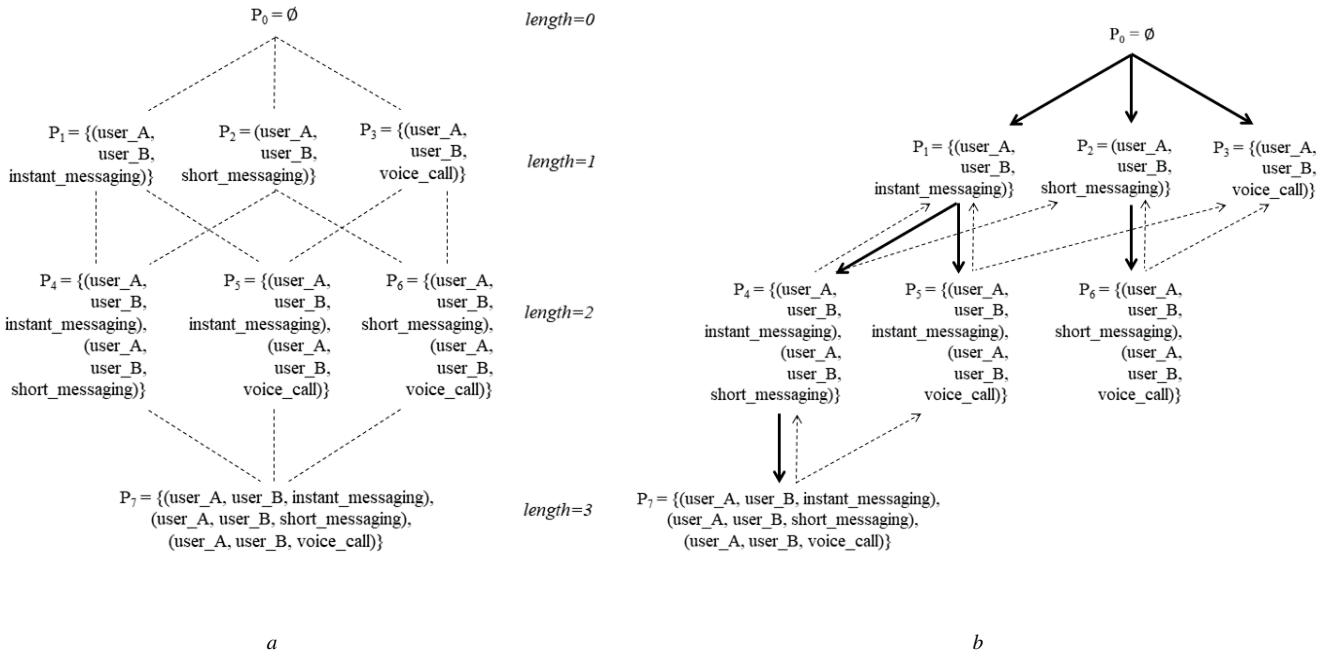


Figure 5: a) Representation of the lattice of subnetworks arranged by generality order $\geqslant$. b) Representation of the SE-Tree used to arrange the lattice on a tree data structure, each vertex of the SE-Tree is a subnetwork.

For instance, the subnetworks

$$P_4 : \{(user\_A, user\_B, instant\_messaging), (user\_A, user\_B, short\_messaging)\}$$
$$P_5 : \{(user\_A, user\_B, instant\_messaging), (user\_A, user\_B, voice\_call)\}$$

are joined to form the subnetwork

$$P_7 : \{(user\_A, user\_B, instant\_messaging), (user\_A, user\_B, short\_messaging), (user\_A, user\_B, voice\_call)\}$$

which is inserted into the SE-Tree in accordance with the preference criteria on the triples (that is, lexicographic order on the labels).

Each vertex is associated to a list (tid list) that stores the time-points in which the triples of the relative subnetwork occur together. These lists are used to identify the frequent (infrequent) subnetworks and allows us to avoid re-scanning the snapshots previously acquired when computing the support. The support values are initialized with the network snapshots collected in the window W', whose initial width is fixed by the user (Figure 4*a*). In particular, the support of the subnetworks of length 1 is simply computed by counting the number of snapshots in W' that include the relative triple, while the support of the subnetworks of length $k+1$ ($k \geq 1$) is determined by computing the set intersection of the tid lists of the two subnetworks at the level k that have been used for the join. The subnetworks whose support values exceeds the threshold $minSUP$ are included in the set $\mathcal{F}_{\mathcal{L}'}$. This operation benefits of the anti-monotonicity of the support, which prevents the exploration of the whole lattice.

During the construction of the SE-tree we do not pose any constraint, the sole condition required is the *connectedness* property of the patterns (see Definition 1). In fact, since we are only interested in sets of triples, in which where any node is connected to any other node through a path, we pose the constraint to join patterns (vertices of the SE-tree) if at least one of them is a subnetwork. The rationale immediately follows the property that we can not obtain subnetworks by joining two patterns that do not represent subnetworks if the patterns have the same prefix. Consequently, patterns which do not represent subnetworks are only kept to "expand" already generated subnetworks at the

next iterations, and not kept and used for further processing steps (e.g. the identification of macroscopic/microscopic changes). Although this aspect could be considered as a limitation of the method, it allows us to do not lose subnetworks which can be obtained from two non-subnetwork patterns and to reduce the size of the lattice (whose size can be a problem in a data stream environment). Moreover, this choice is in line with papers that deal with the problem of identifying an incomplete subset of existing subgraphs [31, 32].

An example of the procedure of construction of the SE-tree is reported in the following and illustrated in Figure 5b. By supposing the level 1 of the lattice already built (Figure 5, length=1), the subnetworks are taken by pairs, in order to build the subnetworks of the level 2. The pairs are created by following the preference criteria on the triples, thus we have (P_1,P_2),(P_1,P_3) and (P_2,P_3). A pair at time is considered: we take P_1 and P_2 and generate P_4, which is appended at the level 2 (Figure 5b, length=2). The "append" operation is completed by associating the tid list to P_4 and computing the support by means of the set intersection of the tid lists of P_1 and P_2. If P_4 is frequent, then it is stored in the list $\mathcal{F}_{\mathcal{L}'}$ (Definition 3.3), which will be used to detect possible macroscopic changes. Subsequently, we take P_1 and P_3 and generate P_5 (Figure 5b, length=2) with the same procedure used for P_4. As two vertices are appended at the level 2, in coherence with the depth-first search, they can be evaluated and joined to build the subnetworks of the level 3. Thus, we join P_4 and P_5, which have the first triple in common, we generate P_7 and append it to SE-tree (Figure 5b, length=3). The subnetwork P_7 is the sole vertex at the level 3, therefore no operation can be performed at that level, then we go back to the preceding level (level 2). There, we find the subnetworks P_4 and P_5 that have been already evaluated and, since no new subnetwork were generated, we go back to the level 1. There, we consider the subnetworks that have not been evaluated before, that is, P_2 and P_3, then we generate P_6 and append it to the level 2. Subsequently, we will consider all the pairs of subnetworks we can create with P_6, that is, (P_4,P_6) and (P_5,P_6), which, recursively, will be evaluated and processed as explained for (P_4,P_5). However, the subnetworks P_4 and P_6 will be not joined because they have not the first triple in common. The same consideration can be done also for P_5 and P_6.

4.2. Update of the lattice

After the construction and initialization, the lattice $\mathcal{L}'$ is continuously updated in the support values as result of the analysis of newly incoming network snapshots. To do this, KARMA uses a data block Π that shifts forward along the stream (sliding window model) and acquires a collection of fixed size of network snapshots (see Figure 4*a*). The data block Π is therefore concatenated to the window W' to build the landmark window $W'' = W' \cup \Pi$, which mirrors the distribution of the occurrences over the network snapshots acquired so far (see Figure 4*b*). Subsequently, KARMA updates the tid lists of the subnetworks by considering the new occurrences present in the data block Π. Such a modification leads to new lattice $\mathcal{L}''$, which is identical to $\mathcal{L}'$ in the structure, but different in the support values of the subnetworks. In fact, the new occurrences may modify the tid lists of the subnetworks with the effect of increasing or decreasing the support, or even making frequent (infrequent) subnetworks that were infrequent (frequent).

To efficiently perform such an update, we fully exploit the implementation of the lattice in form of SE-tree and, in particular, consider the links, due to the generality order, between the subnetworks at the (k+1)-th level and the subnetworks at the k-th level (dashed arrows in Figure 5b). Indeed, any modification done on the tid list of a subnetwork is automatically propagated to the subnetworks at the successive level. Procedurally, for each new data block Π, we only count the occurrences of the triples of the subnetworks at level 1 of the lattice and update the corresponding tid lists. The tid lists of the other levels of the lattice are updated by following (downwards) the links and performing the set-intersection of tid lists. This computation allows us to determine the exact values of the support, while avoiding approximate counts [33]. Finally, to build the set $\mathcal{F}_{\mathcal{L}''}$, we select the subnetworks whose support values exceeds the threshold $minSUP$. This operation benefits of the anti-monotonicity of the support, which prevents the exploration of the whole lattice.

4.3. Detection of macroscopic and microscopic changes

The lattice $\mathcal{L}''$ mirrors the magnitude of the changes to the lattice $\mathcal{L}'$ introduced by the snapshots of the data blocks Π. Thus, we consider the two lattices $\mathcal{L}'$ and $\mathcal{L}''$ and determine the value of $MC(\mathcal{L}', \mathcal{L}'')$ (defined in Section 3.2), in order to quantify the change and establish whether macroscopic changes emerge. To compute $MC(\mathcal{L}', \mathcal{L}'')$, we need the sets of frequent subnetworks $\mathcal{F}_{\mathcal{L}'}$ and $\mathcal{F}_{\mathcal{L}''}$, which are generated when acquiring the snapshots of W' (see Section 4.1) and when acquiring the snapshots $W'' = W' \cup \Pi$ (see Section 4.2), respectively.

Algorithm 2: Detection of emerging subnetworks

```
1  computeEmergingSubnetworks (F_L', F_L'', minGR)
       inputs : F_L', F_L'', minGR
       output: ES s                                   /* the set of microscopic changes */
2      ES s ← ∅
3      foreach P ∈ (F_L'' − F_L') do
4          GR ← sup_W''(P) / sup_W'(P)
5          if GR ≥ minGR then
6              ES s ← ES s ∪ {P};
7          end
8      end
9      foreach P ∈ (F_L' − F_L'') do
10         GR ← sup_W'(P) / sup_W''(P)
11         if GR ≥ minGR then
12             ES s ← ES s ∪ {P};
13         end
14     end
```

Determining the value of $MC(\mathcal{L}', \mathcal{L}'')$ allows us to quantify the variation in the distribution of the occurrences between the window W' and the window $W'' = W' \cup \Pi$. Such a variation has a two-fold nature: the first type corresponds to the evolution of subnetworks which were frequent over W' and which become infrequent over $W' \cup \Pi$ ($|\mathcal{F}_{\mathcal{L}'} - \mathcal{F}_{\mathcal{L}''}|$), while the second type corresponds to the subnetworks which were infrequent over W' and which become frequent over $W' \cup \Pi$ ($|\mathcal{F}_{\mathcal{L}''} - \mathcal{F}_{\mathcal{L}'}|$). This clarifies why we revise the whole lattice of the subnetworks. Indeed, if we had limited the update to the frequent subnetworks only and neglected the infrequent ones, we would have lost the evolution of infrequent subnetworks that become frequent. We deem these change significant

Not all the values of $MC(\mathcal{L}', \mathcal{L}'')$ are deemed as interesting, but only those that exceed the threshold *minMC*. Therefore when the value is less than the threshold *minMC*, we assume that no macroscopic change has been detected. In this case, the analysis continues by keeping the landmark window W'' and processing next data blocks Π. Thus, the old window W'' becomes the new window W' and the old lattice $\mathcal{L}''$ becomes the new lattice $\mathcal{L}'$, as shown in Figure 4c. On the contrary, when the value of $MC(\mathcal{L}', \mathcal{L}'')$ exceeds the threshold *minMC*, an alert for network data changes is raised. This activates a procedure (Algorithm 2) aiming at identifying the subnetworks which exhibit the changes that motivate the macroscopic change.

This procedure is performed by exploring the two lattices and matching for each subnetwork, the support in $\mathcal{L}'$ against the support in $\mathcal{L}''$. More precisely, it checks whether each subnetwork is frequent (infrequent) in $\mathcal{L}'$ and is infrequent (frequent) in $\mathcal{L}''$. Clearly, the subnetworks which are frequent (infrequent) in $\mathcal{L}'$ and remain frequent (infrequent) in $\mathcal{L}''$ denote no evolution and therefore they are excluded because they are not relevant for the description of the change.

However, motivating the macroscopic change with subnetworks, whose support, although relatively stable, "crosses" the *minS UP* threshold could represent a severe limitation of the approach. Reasonably, in these cases, the subnetworks do not provide a relevant contribution to microscopic changes. Indeed, the microscopic changes should be ascribed to subnetworks that specifically exhibit a significant variation of the support. In order to consider this aspect, we evaluate whether Equation (3) is satisfied, that is, the variation of the support of the subnetworks exceeds the threshold *minGR*. Note that Equation (3) takes into account both increase and decrease of the support. The subnetworks which meet these constraints singularly denote emerged microscopic changes.

The procedure (2) that finds the emerging subnetworks is organized in two steps, described in the following:

1. Algorithm 2, lines 3-8: the procedure evaluates each subnetwork $P \in (\mathcal{F}_{\mathcal{L}''} - \mathcal{F}_{\mathcal{L}'})$ which is frequent in $\mathcal{L}''$ and infrequent in $\mathcal{L}'$, and adds to the set of emerging subnetworks only those whose growth-rate $GR_{W'',W'}(P)$ exceeds the user defined threshold *minGR*.
2. Algorithm 2, lines 9-14: the procedure evaluates each subnetwork $P \in (\mathcal{F}_{\mathcal{L}'} - \mathcal{F}_{\mathcal{L}''})$ which is infrequent in $\mathcal{L}''$

and frequent in $\mathcal{L}'$, and adds to the set of emerging subnetworks only those whose growth-rate $GR_{W',W''}(P)$ exceeds the user defined threshold *minGR*.

Finally, the analysis re-starts by initializing the window W' with the network data which have determined the macroscopic change (i.e., $W' = \Pi$). The lattice does not change, but the support values of the subnetworks are re-initialized by only considering the occurrences present in the network snapshots of the window W' (see Figure 4d).

4.4. Time complexity analysis

We evaluate the time complexity of KARMA by considering the worst-case scenario. As for many frequent pattern mining algorithms, we will conclude that the worst-case complexity is far not compatible with the real-world applications. For this reason, we also analyze the average-case time complexity of the algorithm by taking advantage of previous theoretical results.

Let n be the number of nodes (i.e. $n = |N|$), m be the number of edge labels (i.e. $m = |\mathcal{E}|$), W' be the landmark window and Π be the data block of network snapshots such that $W'' = W' \cup \Pi$. The computational complexity of KARMA depends on: i) the cost of constructing and updating the lattice of subnetworks over the data stream, ii) the cost of computing the value of *MC*, iii) the cost of identifying the emerging subnetworks.

i) Since the algorithm needs to explore the lattice of subnetworks, the total number of frequent subnetworks is bounded by the cardinality of the power set of all the possible triples $\mathcal{P}(N \times N \times \mathcal{E}) = 2^{|N|\cdot|N|\cdot|\mathcal{E}|} = 2^{n^2 m}$, therefore $O(2^{n^2 m})$. It is noteworthy that this size is in line with usual worst-case scenario in frequent pattern mining. However, since we (as all the frequent pattern mining algorithms) use the minimum support threshold, this size is, in practice, much smaller. Moreover, KARMA is able to generate only (connected) subnetworks. Supposing that each node is connected to at most other $k < n$ nodes in a network snapshot, then the total number of frequent subnetworks that can be generated by KARMA is, in the worst-case, $O(2^{nkm})$. Obviously, this size affects the time complexity. Indeed, when joining two subnetworks of length i to form a subnetwork whose length is $(i + 1)$, we should compute the intersection of their respective tid lists of size $O(|W''|)$, whose cost is $O(2^{nkm})$, resulting in a total cost of $O(2^{nkm} \cdot |W''|)$. However, KARMA adopts a different solution: the intersections are computed for each data block Π and, therefore, incrementally maintained. Thus, when the new data block Π arrives, the old tid lists are adapted by accounting for the snapshots of Π. As a consequence, the cost of the intersection is $O(|\Pi|)$, resulting in an overall cost of $O(2^{nkm} \cdot |\Pi|)$, which can be significantly better than $O(2^{nkm} \cdot |W''|)$ because $|\Pi| \leq |W''|$

ii) The computational cost of computing the value of MC, in theory, is proportional to the size of $\mathcal{F}_{\mathcal{L}'}$ (relative to W') and $\mathcal{F}_{\mathcal{L}''}$ (relative to W'') because it requires the computation of three cardinalities: $|\mathcal{F}_{\mathcal{L}'} - \mathcal{F}_{\mathcal{L}''}|$, $|\mathcal{F}_{\mathcal{L}''} - \mathcal{F}_{\mathcal{L}'}|$ and $|\mathcal{F}_{\mathcal{L}'}|$. However, these three cardinalities can be computed avoiding set differences during the update of $\mathcal{L}'$. This is done by simply counting subnetworks whose updated support "crosses" the minimum support threshold value. Therefore, the cost of computing the value of *MC* is $O(1)$.

iii) Since we are only interested in computing the GR of frequent (infrequent) subnetworks that become infrequent (frequent), the identification of emerging subnetworks has a computational cost that is proportional to new_{f} and to new_{if}, where $new_{\mathrm{f}} = |\mathcal{F}_{\mathcal{L}''} - \mathcal{F}_{\mathcal{L}'}|$, $new_{\mathrm{if}} = |\mathcal{F}_{\mathcal{L}'} - \mathcal{F}_{\mathcal{L}''}|$.

Therefore, the worst-case time complexity of the entire process executed when a new block arrives is:

$$O\left(\underbrace{2^{nkm} \cdot |\Pi|}_{\text{update cost}} + \underbrace{new_{\mathrm{f}} + new_{\mathrm{if}}}_{\text{extraction of emerging subnetworks}}\right)$$

Since $new_{\mathrm{f}} + new_{\mathrm{if}} \ll 2^{nkm}$, because new_{f} and new_{if} refer to subnetworks contained in the lattice, the final complexity is $O(2^{nkm} \cdot |\Pi|)$ or, in other terms, the time complexity of KARMA is linear in the size of the lattice and linear with respect to the size of the arriving block.

As we can see, the exponential complexity in the worst-case scenario is rather incompatible with real-world applications. In fact, the combinatorial explosion of frequent subnetworks can easily violate the streaming constraints imposed by the problem that KARMA solves. In the average-case scenario, the overall amount of processing that needs to be done depends on the content of the window $W'' = W' \cup \Pi$. In fact, each snapshot of the window can contribute to altering the relative frequencies of subnetworks and, therefore, the update of the lattice.

A common setting that has been proven useful when studying the average-case is that of the random shopper that randomly fills any transaction with items. In the case of KARMA, the analysis is based considering two assumptions also considered in [34]:

- The random shopper assumption. W'' can be seen as a random database of snapshots from nkm triples in which each snapshot is produced by the same probabilistic source. The fact that a triple belongs or not to a snapshot can be modeled as an independent random variable which follow a Bernoulli distribution, with $\theta \in (0, 1)$. As a consequence, the probability $\mathbb{P}(P)$ that a snapshot include a subnetwork P exponentially decreases with the size of the subnetwork ($\exists M > 0$ such that $\mathbb{P}(P) \leq M \cdot \theta^{|P|}$).
- The linear frequency threshold assumption. Because the minimum (relative) support threshold $minSup \in [0, 1]$ is constant we have that the corresponding minimum absolute support threshold r grows linearly with respect to the size of the window. Thus, resulting in $r \propto minSup \cdot |W''|$, when $|W''| \to \infty$.

When these assumptions hold, [34] proved that the number of frequent patterns is polynomial with respect to the number of items. So, instead of having $O(2^{nkm})$ frequent subnetworks, KARMA considers only $O(c(nkm)^s)$ where $s = \lfloor \frac{log(minSup)}{log(\theta)} \rfloor$ and $c = \frac{1}{s!}$ in the average-case. The number of frequent (infrequent) subnetworks that become infrequent (frequent) is still dominated by $O((nkm)^s)$, so the cost of computing the value of MC and the cost of identifying the emerging subnetworks remains unchanged.

Therefore, the average-case time complexity of the entire process executed when a new block arrives is:

$$O\left(\underbrace{(nkm)^s \cdot |\Pi|}_{\text{update cost}} + \underbrace{new_{\text{f}} + new_{\text{if}}}_{\text{extraction of emerging subnetworks}}\right)$$

This means that the final complexity is $O((nkm)^s \cdot |\Pi|)$ in the average-case or, in other terms, the time complexity of KARMA is polynomial in the number of triples with an exponent that depends on the minimum support threshold. Furthermore, the time complexity is still linear with respect to the size of the arriving block.

This kind of analysis depends on the nature of the data considered and do not depend on a specific frequent pattern mining algorithm. By doing this kind of analysis the average cost of the KARMA algorithm is asymptotically the same, regardless of the frequent pattern mining strategy used during the update of the lattice.

In conclusion, considering both the worst-case and the average-case scenario, we can state that KARMA does not add further time complexity (under the $O(\cdot)$ operator) to that of the classical frequent pattern mining problem, while addressing a more complex data mining task which comprises that of frequent pattern mining.

5. Experiments

In order to empirically evaluate KARMA, we performed experiments that aim at qualitatively and quantitatively evaluating the changes expressed by the lattice (representing the whole network) and by the emerging subnetworks. The experiments are organized along the following perspectives:

- we study the influence of the parameters *minGR* and *minMC* on the macroscopic changes, microscopic changes and running times.
- we present the results of a comparative evaluation between KARMA and the algorithms proposed in Berlingerio et al. [14], Loglisci et al. [18] and Koh [16], which analyze the evolution of the network with frequent pattern-based methods. The algorithm described in Koh [16], specifically, works on data streams, similarly to our approach.
- we argue the usefulness of the proposed solution in real-world applications with three practical scenario.

Additionally, we evaluate the proposed solution with three variants designed by replacing the algorithm of discovery of subnetworks (described in Section 4.1) with a method of subgraph mining [35] and a method of itemset mining in data stream [36]. The evaluation of these variants is reported as Appendix to this manuscript.

As we will observe, the experimental results show that KARMA *i)* requires less time consumption, as compared to the competitors, *ii)* is sensitive to the changes of the whole network and *iii)* identifies the subnetworks that better express the evolution of the network.

The experiments are run on an Intel i7 64bit @3.4 GHz desktop running Windows. The source code of the system, the dataset and some resulting subnetworks are publicly available on Bitbucket[1,2]. The datasets used have time-stamped networks and are stored on the desktop machine as data files. They are converted into data streams by taking the temporal order on the time-stamps as the order of the network data stream. In the following subsections, we first present the used datasets and then provide details on the four experiments and obtained results.

5.1. Dataset Description

We performed experiments on five real-world network data streams with different characteristics. In Table 2, we report the basic statistics and network-based indexes computed by considering each dataset as a stream of network snapshots. More specifically, we report the total number of snapshots and the minimum and the maximum values of the i) number of nodes, ii) number of edges, iii) node degree, iv) diameter and v) density over the snapshots.

Table 2: Basic statistics and network-based indices of the datasets. Values are aggregated over time-points.

dataset	total number of snapshots	max/min number of nodes over the snapshots	max/min number of edges over the snapshots	max/min node degree over the snapshots	max/min diameter over the snapshots	max/min density over snapshots
KEDS	11070	36/2	91/1	40/1	9/1	5/0.079
NODOBO	41344	26/2	2600/1	2076/1	8/1	72.3/0.35
NOAA	7670	1738/1418	3917/2601	8/1	76/44	0.0028/0.0022
WikiTalk	2185	18344/1	28017/1	14400/1	45/1	45/1
MAWI	51809	360/2	343/1	231/1	1/1	5/0.0053

The first dataset (*KEDS*) concerns the geographic-social-political scenario described on the news reports on the Gulf region and collects data on the social and political relationships among nations and world-wide organizations. The process of construction of these data is illustrated in Schrodt et al. [37], while the dataset is available at `http://eventdata.parusanalytics.com/data.dir/levant.html`. In this work, as in [38], we consider this dataset as a network, where nations and world-wide organizations represent the nodes and social and political relationships correspond to the edges between the nodes. In KEDS, the labels of the nodes and of the edges are 208 and 20, respectively. The dataset is collected day by day from April 1979 to July 2009, therefore the time-point representation is in the format year/month/day (one time-point represents one day). The goal is to identify particular variations in the flow of the news reports and associate them to actual changes in social and political relationships among nations and world-wide organizations.

The second dataset (*NODOBO*) concerns a communication network and contains telecommunication transactions gathered during a study of the mobile phone usage of 27 students of a Scottish state high-school, from September 2010 to February 2011 [39]. The students communicate by phone calls, text messages (sms) and bluetooth connection. This dataset includes 13035 phone call records, 83542 text message records, 5292103 bluetooth records, and it is accessible at `http://nodobo.com/release.html`. In particular, we have information on the duration of the phone calls, length of the text messages and type of device used in the bluetooth connections. When building the network, the nodes represent the students, while the edges represent the different modalities of communication. The labels of the edges are 11: 5 different labels for the communication based on phone calls (*short_call*, *short_medium_call*, *medium_call*, *medium_long_call*, *long_call*); 5 different labels for the communication based the length of text messages (*short_length*, *short_medium_length*, *medium_length*, *medium_long_length*, *long_length*); one label for the presence of the bluetooth connection. The labels for the calls and the messages were generated by applying an equal-width discretization technique with 5 bins to the values of the duration of the phone calls and length of the text messages respectively. One time-point represents a time-interval of five minutes. The goal is to identify changes in the way

[1] `https://bitbucket.org/netminerteam/karmaalgorithm`
[2] `https://bitbucket.org/netminerteam/datasets`

communications are performed in terms of frequency, type and size (minutes for calls, length for messages, type of communication, individuals with who interacting).

The third dataset (*NOAA*) was developed in the context of the Reanalysis project by the National Center for Environmental Prediction and the National Center for Atmospheric Research. The project aimed at providing new atmospheric analysis by gathering daily measurements of various meteorological quantities (such as, relative humidity and air temperature) by means of geo-localized sensors equally distributed over space [40]. In this work, we built the network with the measurements of relative humidity of the time-interval January, 1st 1990 - December, 31st 2010, recorded daily on an area that roughly covers North-Central America. The nodes of the network represent the sensors, while the edges are nominal values denoting the relative humidity values measured on the two linked sensors. In particular, an equal-width discretization technique with 10 bins was applied to the values of the relative humidity, in order to map one time-stamped value into one bin. The bins generated by the discretization are $(-\infty, 10]$, $(10, 20]$, $(20, 30]$, $(30, 40]$, $(40, 50]$, $(50, 60]$,$(60, 70]$,$(70, 80]$, $(80, 90]$ and $(90, +\infty)$. Then, the label of an edge is defined as a string indicating the coordinates (latitude and longitude) of the connected nodes and their bins of humidity. For instance, to denote the edge between the node 15.0_275.0 (latitude 15.0, longitude 275.0) and the node 15.0_280.0 (latitude 15.0, longitude 280.0), which have been mapped into the bins $(10, 20]$ and $(30, 40]$ respectively, we use the label $bins\_(10, 20]\_\_(30, 40]$. The goal is to *i)* identify substantial meteorological changes of the geographic area under investigation and **ii)** find spatial regions from which the meteorological changes may originate *iii)* determine the temporal collocation (e.g., days) in which the changes occur. The dataset was downloaded from `https://coastwatch.pfeg.noaa.gov/erddap/griddap/esrlNcepRe.html` on March 21st, 2017.

The fourth dataset (WikiTalk) concerns the network of interactions among the authors of Wikipedia, a free encyclopedia collaboratively written by volunteers around the world. Each author possesses a talk page, that (s)he or other users can edit in order to communicate and discuss updates to various articles on Wikipedia. The dataset is available at `https://snap.stanford.edu/data/wiki-Talk.html`. In this work, this dataset forms a network where authors are represented as nodes and the edit performed by an author towards the wikipedia talk page of another author is denoted as an edge between two nodes. In WikiTalk, the number of labels of the nodes and of the edges are 1140141 and 1, respectively. The dataset is collected day by day from December 2001 to January 2008, therefore the time-point representation is in the form year/month/day (one time-point represents one day). The goal is to identify interesting variations in the flow of informations reported among authors with respect to the content of Wikipedia.

The fifth dataset (MAWI) was developed in the context of the MAWI Project by the Measurement and Analysis on the WIDE Internet Working Group. The MAWI Project carried out network traffic measurement, analysis, evaluation, and verification in order to evaluate whether the network behaves as it was designed, and to learn from unexpected behaviors. In this work we used a portion of the whole dataset provided by the MAWI Working Group, that is only the traffic data monitored, in form of IPv6 packets sent over the network, by the sampling point D in the interval January, 25th 2005 -January, 31th 2005. The dataset is available at `http://mawi.wide.ad.jp/mawi/samplepoint-D/2005/`. In particular, for each packet observed we have information about the source IPv6 address, the destination IPv6 address and about the communication protocol (e.g.: IPv6, TCP, UDP, ICMP, etc.). When building the network the nodes represent the IPv6 addresses, while the edges represent the communication protocol between two devices. The labels of the edges are 2745, they simply represent well-known communication protocols such as ICMP, IPv6 and, in the case of UDP and TCP, the label also includes the port number used during the transmission. For example, the pair (TCP,80) indicates an HTTP packet. One time point represents a time-interval of five seconds, this leads to a very high number of snapshots observed.

5.2. Influence of the input parameters

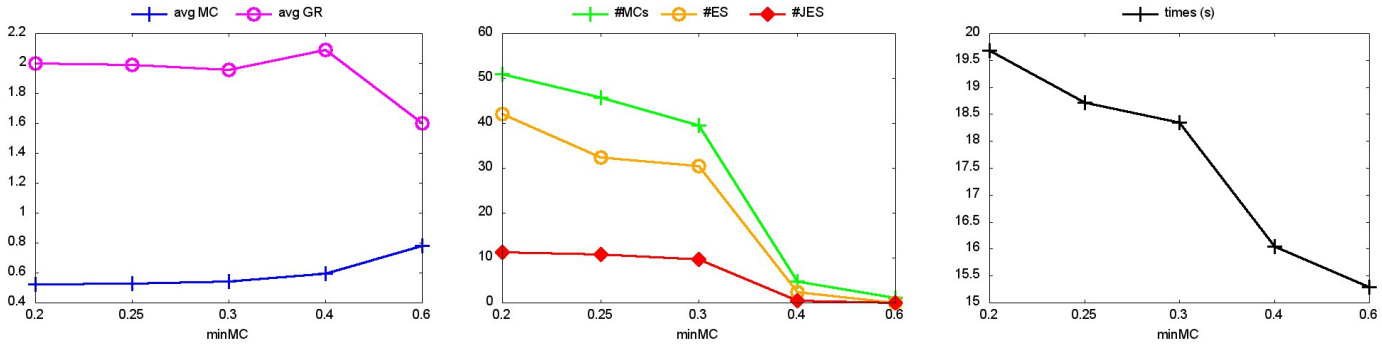
In the first experiment, we test the influence of the input parameters on the running times and quantitative characteristics of the macroscopic and microscopic changes. We manually tuned two input parameters, the minimum threshold of macroChange (*minMC*) and minimum threshold of growth-rate (*minGR*), and collected the results in terms of the statistics listed in Table 3. More precisely, we varied, in turn, the value of one threshold and fixed the value of the other one.

Table 3: Collected statistics

times	Running times (in seconds)
#MCs	Number of detected macroscopic changes (cases in which the threshold inequality $MC > minMC$ if satisfied)
avg MC	Average *MC* value of the cases in which the threshold inequality if satisfied ($MC > minMC$)
avg GR	Average value of the growth-rate of the emerging subnetworks for each macroscopic change. This statistic is computed by excluding emerging subnetworks whose growth-rate is equal to infinite
# ES	Total number of microscopic changes associated to emerging subnetworks with value of growth-rate lower than infinite
# JES	Total number of microscopic changes associated to emerging subnetworks with value of growth-rate equal to infinite

The choice of the appropriate values of the input parameters has no general solution and not always can be handled with automatic techniques. Very often, it requires preliminary computations, exploratory executions and background knowledge of the specific domain. On the other hand, the use of parameters is operatively beneficial for the user because (s)he can focus on the stronger variations while discarding the weaker ones both at the level of the macroscopic and microscopic changes.

The results are reported in Figures 6 and 7 and are obtained as average on the values resulting from the five datasets. In this experiment, the initial width of W' is set to 365, 60, 180, 30 and 60 snapshots for *KEDS*, *NODOBO*, *NOAA*, *WikiTalk* and *MAWI*, respectively.

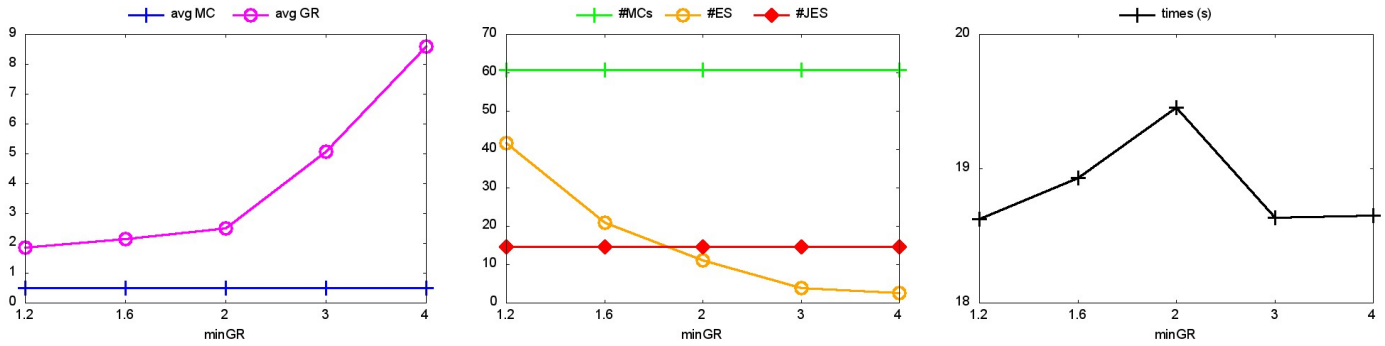


a *b* *c*

Figure 6: Average results produced when tuning *minMC* ($minSUP$=0.1, $minGR$=2). To visualize the results at the same scale, before the average computation, the original values of *#ES* for KEDS, NODOBO, NOAA, WikiTalk and MAWI have been multiplied by 10^{-1}, 10^{-5}, 10^{-3}, 10^{-4} and 10^{-1}, respectively; the values of *#JES* for NODOBO and WikiTalk have been multiplied by 10^{-5} and 10^{-1}, respectively; the values of *#MCs* of NODOBO and NOAA have been multiplied by 10^{-1}; the values of *times* for NODOBO, NOAA, WikiTalk and MAWI have been multiplied by 10^{-1}.

Figures 6*a* and 6*b* report the statistics concerning macroscopic changes and microscopic changes when tuning *minMC*, while figure 6*c* reports the running times. As we can observe, the value of *minMC* affects the results of the several procedures of the algorithm KARMA. Specifically, when *minMC* increases, we require that the algorithm detects macroscopic changes with relatively larger values of *MC*, which excludes macroscopic changes with relatively small values of *MC*. The first result is the increasing tendency of *avg MC*. The second result is the drop of *#MCs*, which is due to the reduced number of operations of re-initialization of the lattice. In such cases, the newly incoming network snapshots do not provide a number of occurrences sufficient to determine a strong variation of the subnetworks of $\mathcal{F}_{\mathcal{L}''}$ with respect to the subnetworks of $\mathcal{F}_{\mathcal{L}'}$. The third result is the decreasing tendency of the running

times, explained with the reduced number of executions of the procedure of construction of the lattice (Section 4.1) and procedure of detection of microscopic changes (Section 4.3). Indeed, the algorithm of detection of emerging subnetworks will be activated only when there is a strong variation between the set $\mathcal{F}_{\mathcal{L}''}$ and the set $\mathcal{F}_{\mathcal{L}'}$. This explains the drop of $\#ES$ and $\#JES$.



a *b* *c*

Figure 7: Results produced when tuning *minGR* (*minSUP*=0.1, *minMC*=0.25). To visualize the results at the same scale, before the average computation, the original values of $\#ES$ for KEDS, NODOBO, NOAA, WikiTalk and MAWI have been multiplied by 10^{-1}, 10^{-5}, 10^{-3}, 10^{-4} and 10^{-3} respectively; the values of $\#JES$ for NODOBO and WikiTalk have been multiplied by 10^{-5} and 10^{-1}; the values of $\#MCs$ of NODOBO, NOAA and MAWI have been multiplied by 10^{-1}; the values of *times* for NODOBO, NOAA, WikiTalk and MAWI have been multiplied by 10^{-1}.

Figures 7*a* and 7*b* show the effect of the parameter *minGR* on the macroscopic changes and microscopic changes, while figure 7*c* reports the running times. As expected, the parameter *minGR* has effect only on the microscopic changes, while has no influence on the statistics *avg MC*. In particular, we observe that the value of # *ES* (number of microscopic changes) decreases and the value of *avg GR* (variation of frequency exhibited by the microscopic changes) increases when *minGR* increases. This is not surprising because the strongest microscopic changes are exhibited by a smaller set of subnetworks. Clearly, the values of # *JES* do not show any variation since they only take the emerging subnetworks with infinite growth-rate into account. Accordingly to the study of the time complexity (see Section 4.4), the procedure to detect emerging subnetworks has small influence on the time consumption, regardless of the value of *minGR*. This explains the constant behavior of Figure 7*c*.

5.3. Comparative evaluation

The comparative evaluation has been performed between the algorithm KARMA and the algorithms described in Berlingerio et al. [14] (afterwards GERM), Loglisci et al. [18] (afterwards RANGER) and Koh [16] (afterwards CD-TDS). Since GERM and RANGER work on network data, but not data streams, whereas CD-TDS works on data streams, but not on network data, some adaptations were necessary. Before analyzing the obtained results, we report some technical details of the competing algorithms and of the needed adaptations.

GERM discovers patterns, called evolution rules, which characterize the most frequent evolutions of the network over time. An evolution rule reflects the same evolution in its multiple occurrences over time. In this case, we simulate the input stream to generate a cumulative graph, which is given as input to GERM.

RANGER discovers two kinds of patterns, that is, change patterns and change chains. While change patterns capture the same evolution exhibited by different network snapshots between two consecutive windows, change chains are successions of such evolutions on the analyzed temporal axis and, thus, characterize the most frequent evolutions over a succession of fixed windows. In this case, we simulate the stream by aggregating data by time window.

CD-TDS detects local changes (changes of the frequency of individual items) and global changes (changes of co-occurrences of items). To perform a fair comparison, we modified CD-TDS in two directions. First, we used SE-trees of triple sets, instead of prefix trees of items, in order to make CD-TDS able to handle graph data rather than transactional data, as in its original version. Second, in the evaluation local changes and global changes are assimilated to microscopic changes and macroscopic changes, respectively.

As in the previous analysis, we chose to perform the comparative evaluation on the dataset KEDS, because it has the longest time-interval. The analysis of the results was performed along three dimensions: the running times, the number of discovered patterns and the average number of windows between two significant changes of the network. While the number of discovered patterns refers to the topological regularities which exhibit changes, the average number of windows between two significant changes indicates the number of windows in which macroscopic changes are identified. To guarantee a fair comparison, we considered the same temporal granularity for all four approaches. In particular, we considered two reference widths for time-windows: 90 and 180 days. On the basis of such reference widths we have organized the input data for each algorithm as follows. In our approach, the initial width of W' coincides with the width of the block and both cover 90 and 180 days, that is, 90 and 180 time-points. In the case of GERM, the data associated to each time-point are obtained by collecting the edges observed in the periods of 90 and 180 days. For RANGER and CD-TDS, all the windows have a width equal to 90 and 180 days. Experiments were performed by varying the threshold $minSUP$, which is common to all four algorithms. We report the averages of the results obtained with both widths.

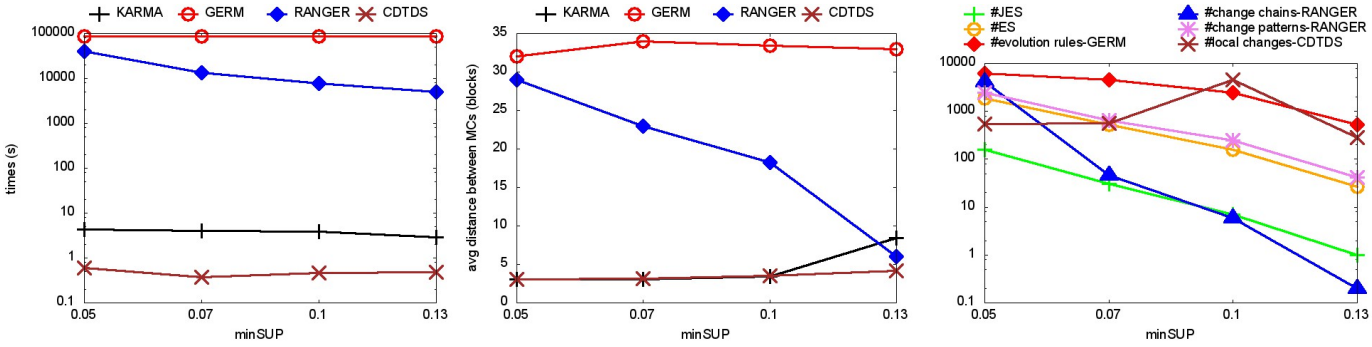


Figure 8: Comparison with three competitors by considering the running times (a), average size of windows separating macroscopic changes (b) and number of patterns (c) while tuning the threshold $minSUP$ ($minGR$=2 and $minMC$=0.1).

In Figure 8*a*, we can see that the running times of KARMA are significantly smaller than those of the algorithms GERM and RANGER and are comparable with those of CD-TDS. In particular, there is a difference of (at least) two orders of magnitude (data are plotted in log scale) with GERM and RANGER. For GERM, *times* remains unchanged, since we set the maximum running time for the experiments to 24 hours of uninterrupted execution. This result is motivated by a different algorithmic choice. More precisely, GERM operates directly on the cumulative graph, from which it mines frequent subgraphs that express the evolutions. On the contrary, our approach does not look for changes directly in the network snapshots, but acts on the set of patterns. As for RANGER, we motivate the difference with the choice of modeling data and patterns with the first-order logic, that, as previously stated, can turn out to be time-consuming. In the case of CD-TDS, the time consumption is slightly smaller than KARMA, but of the same order of magnitude, because CD-TDS requires only a local change (variation of the frequency of just an item) to slide forward the window of the stream.

In Figure 8*b*, we evaluate the average distance between two macroscopic changes. In the case of GERM, we consider the average length of the evolution rules, since the rule ends when it is not possible to extend it to other windows (i.e., it is not possible to match the rule with the new snapshots of the network). Similarly, in the case of RANGER, we consider the average length of the change chains, since the chain ends when it is not possible to extend it to other windows (i.e., it is not possible to match the last pattern of the chain with the new snapshot of the network). In the case of CD-TDS, we consider the average length of the windows when: i) a global change is detected, that is, when two SE-trees (lattices), built on two time-windows, are significantly dissimilar with respect to the statistical test defined in Koh [16] or ii) when a local change is detected. In KARMA, we consider the average length of the windows when a macroscopic change is detected. By comparing the results (expressed in months), we can see that the three algorithms have different behavior. As $minSUP$ increases, in KARMA we observe that the distance between macroscopic changes increases. In RANGER, we see a drop, which is due to the relatively short

change chains generated from a smaller set of change patterns. As for GERM, the length of the evolution rules remains quite constant because they cover the same changes which can occur in different sequences of windows. We should also consider that the experiments of GERM have been interrupted after 24h of execution. As for the comparison with CD-TDS, we observe that the average distances are comparable, which is quite expected, considering that global changes (CD-TDS) and macroscopic changes (KARMA) are determined in a different way, but starting from the same data structures (SE-trees) which potentially express the same information.

In Figure 8*c*, we report the number of patterns which model the changes (plotted in log scale). The number of *evolution rules* (in GERM) is several orders of magnitude higher than the number of emerging subnetworks $\#JES$ and $\#ES$. While, the number of *change chains* and *change patterns* in RANGER is generally of the same order of $\#JES$ and $\#ES$. The huge number of results produced by GERM is due to the size of the search space in which the evolution rules are discovered. Indeed, they are built in a combinatorial way by combining all the changes captured over the complete succession of time-points. On the contrary, the search space of the emerging subnetworks is built only from the windows in which macroscopic changes are detected. As a confirmation, if we consider separately the time-windows on which RANGER discovers changes and we evaluate only the number of change patterns (#*change patterns*), this number is comparable with $\#ES$. As for CD-TDS, the number of local changes is generally larger than microscopic changes of KARMA (more evident on $\#JES$) because CD-TDS identifies any (statistically significant) variation on the frequency of single items (triples), which is an event more recurrent than the variation of frequency of subnetworks (triple set). This allows us to clarify that, while the algorithm CD-TDS discovers changes that regard single edges and nodes, KARMA characterizes the evolution at the level of subgraphs, which, in many applications, can provide more abstract and more general information.

5.4. KARMA at work

In this section we show the applicability of KARMA for the analysis of a real world network by discussing some discoveries and commenting their actionability with respect to facts and events occurred in the domain.

The capacity of analyzing complex data in a real-time setting collocates KARMA in the category of tools designed for problems of monitoring of evolving complex phenomena. Here, we suggest to visualize the changes detected by KARMA in form of time-series, in order to support a visual inspection by experts. A practical example is illustrated in Figure 9 which shows a projection of the macroscopic changes as a function of time. The amplitude of the points of the time-series indicates the number of corresponding microscopic changes, so the higher the amplitude, the higher the number of involved emerging subnetworks, the higher the impact of the changes on the whole network. Those points can be of high interest for the monitoring of the phenomena and can suggest some clues for experts. To ease the inspection, in Figure 9, we highlight the time-point in which the macroscopic change starts.

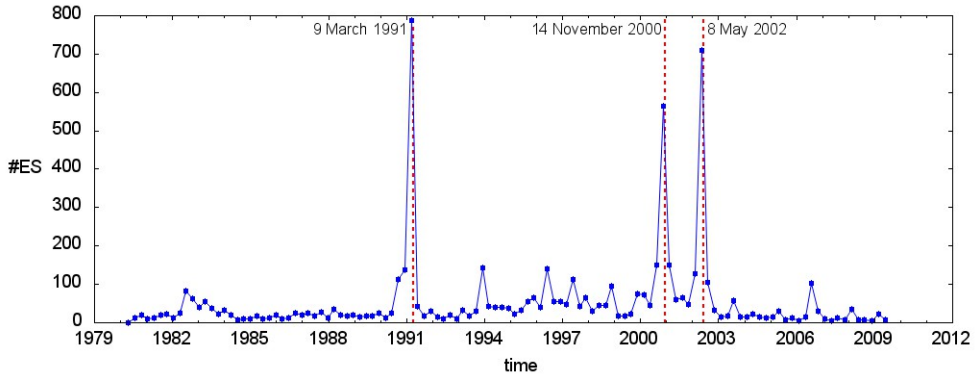


Figure 9: Distribution of the microscopic changes #ES discovered from the KEDS dataset as a function of time. The values of the #ES has been multiplied by 10^{-1}.

In the domain of KEDS, it is possible to use KARMA to analyze time-lines from news broadcasting web-sites. Time-lines have become a common solution to explore and navigate (in a simple and effective way) real-world processes and phenomena through temporally-related key events [41]. A microscopic change, identified by KARMA,

may denote a single real-world event by specifying the involved entities, the relationships existing among these entities and their temporal collocation.

We remind that KEDS collects data on the social and political relationships among nations and world-wide organizations extracted from news reports concerning the Gulf area. By observing Figure 9, we can clearly see three peaks, each characterized by more than 500 microscopic changes. The first one (begin of 1991) can clearly be associated to the First Gulf War (Operations Desert Shield and Desert Storm). The second peak corresponds to the Second Intifada (started in September 2000) and the third peak (late 2001 - early 2002) corresponds to the invasion of Afghanistan occurred after the September 11 attacks. From the picture, it is also possible to identify other peaks. For example, there is a peak in 2006 corresponding to the 2006 Lebanon War.

By inspecting the emerging subnetworks of the peak "March 9th 1991", we find the following:

P_1: {(iraq,kuwait,fight), (usa,iraq,fight), (ussr,usa,disapprove)}

$GR_{[1990\ Dec\ 12,\ 1991\ Jun\ 7],\ [1990\ Dec\ 12,\ 1991\ Mar\ 8]}(P_1) = 2.0$
$macroChange = 0.88$
$minSUP = 0.02$, $minMC = 0.25$ and $minGR = 2$

It participates to a macroscopic change of the network, which starts on March 9th 1991 (the time-point after the window [1990 Dec 12, 1991 March 8]) and terminates on June 7th 1991. This change is quantified with the value of 0.88 (in the range [0,1]) and, as we remind, it refers to the variation of the set of the frequent subnetworks and variation of the infrequent subnetworks, which include also the emerging subnetworks. At a microscopic level, the emerging subnetwork P_1 has a decrease of the frequency by a factor of 2. We evaluated qualitatively the validity of some discoveries with available time-line services. As for P_1, we exploited the Timeline Archive offered by Information Please encyclopedia and, in particular, one event mentioned in the time-line for "Persian Gulf War" [3]:

"1991 Bush wins congressional approval for his position with the most devastating air assault in history against military targets in Iraq and Kuwait (Jan. 16). He rejects a Soviet-Iraq peace plan for a gradual withdrawal that does not comply with all the UN resolutions and gives Iraq an ultimatum to withdraw from Kuwait by noon Feb. 23 (Feb. 22). The president orders the ground war to begin (Feb. 24). In a brilliant and lightning-fast campaign, U.S. and coalition forces smash through Iraq's defenses and defeat Saddam Hussein's troops in only four days of combat. Allies enter Kuwait City (Feb. 26). Iraqi army sets fire to over 500 of Kuwait's oil wells as final act of destruction to Kuwait's infrastructure. Bush orders a unilateral cease-fire 100 hours after the ground offensive started (Feb. 27). Allied and Iraqi military leaders meet on battlefield to discuss terms for a formal cease-fire to end the Gulf War. Iraq agrees to abide by all of the UN resolutions (Mar. 3). The first Allied prisoners of war are released (Mar. 4). Official cease-fire accepted and signed (April 6). 532,000 U.S. forces served in Operation Desert Storm. There were a total of 147 U.S. battle deaths during the Gulf War, 145 nonbattle deaths, and 467 wounded in action. "

and one event mentioned in the time-line for "Iraq" [4]:

"1991 [...] Formal cease-fire is signed. Saddam Hussein accepts UN resolution agreeing to destroy weapons of mass destruction and allowing UN inspectors to monitor the disarmament (April 6). A no-fly zone is established in Northern Iraq to protect the Kurds from Saddam Hussein (April 10). UN weapons inspectors report that Iraq has concealed much of its nuclear and chemical weapons programs. It is the first of many such reports over the next decade, pointing out Iraq's thwarting of the UN weapons inspectors (July 30).. "

Thus, P_1 depicts the scenario successive to the cease-fire (April 6) of the First Gulf war, in which the number of news reports describing the conflict or state of war diminishes.

[3] https://www.infoplease.com/history-and-government/1900-1999-ad-world-history/persian-gulf-war-jan-16-1991-april-6-1991
[4] https://www.infoplease.com/spot/iraq-timeline

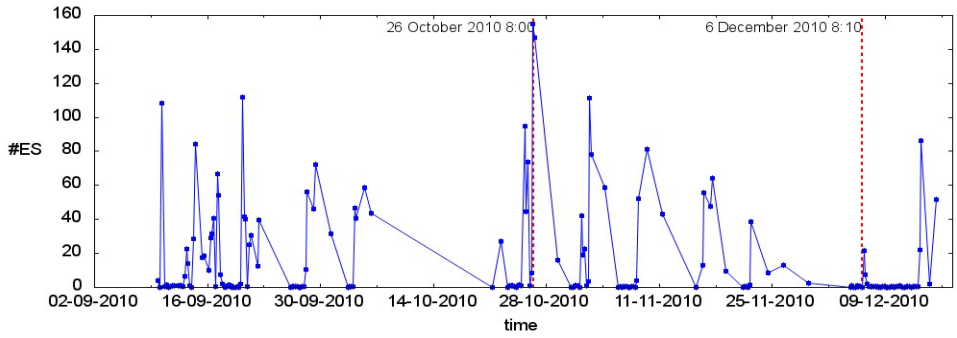


Figure 10: Distribution of the microscopic changes #ES discovered from the NODOBO dataset as a function of time. The values of the #ES has been multiplied by 10^{-3}.

As for the analysis of NODOBO, in Figure 10, we see a succession of points with a decreasing trend, which has the peak on October 26th, 2010 and the lowest number of microscopic changes on December 6th, 2010. To give a practical interpretation to this behavior, it is useful to say that in Scottish state high-schools there is a holiday period which covers the second and third week of October, thereafter the school activities continue. So, the projection of Figure 10 reveals that, when the school activities resume, there is high variability (many microscopic changes) in the modalities of communication, which, as time goes by, tends to decrease. This may provide indications on the use of mobile phone of the students, which can be exploited, for instance, to plan the school policies and to improve the mobile network services in the area.

Among the emerging subnetworks associated to the peak, we find the following:

P_2: {(student_id14,student_id0,bluetooth),
(student_id18,student_id0,bluetooth), (student_id2,student_id0,high_length)}

$GR_{[\text{2010 Oct 25-22:00, 2010 Oct 26-2:55}],\,[\text{2010 Oct 25-22:00, 2010 Oct 26-7:55}]}(P_2) = 2.0$
$macroChange = 0.94$
$minSUP = 0.02$, $minMC = 0.25$ and $minGR = 2$

It is involved in the strongest macroscopic change (quantified as 0.94), which starts on October 26th 2010 at 3:00 (the time-point after the window [2010 Oct 25-22:00, 2010 Oct 26-2:55] and terminates on October 26th 2010 at 7:55. Specifically, P_2 denotes the doubling (GR=2.0) of the occurrences of the subnetwork {(*student_id14, student_id0, bluetooth*), (*student_id18, student_id0, bluetooth*), (*student_id2, student_id0, high_length*)} from the window [2010 Oct 25-22:00, 2010 Oct 26-2:55] to the window [2010 Oct 25-22:00, 2010 Oct 26-7:55]. On the contrary, (we verified) this specific change does not appear in the set of emerging subnetworks discovered between the successive landmark windows [2010 Dec 05-22:10, 2010 Dec 06-3:05] and [2010 Dec 05-22:10, 2010 Dec 06-8:05], which may indicate that the interaction among the three students becomes stable, that is, there is no relevant variation on the number of occurrences in the modalities of communication of those three students.

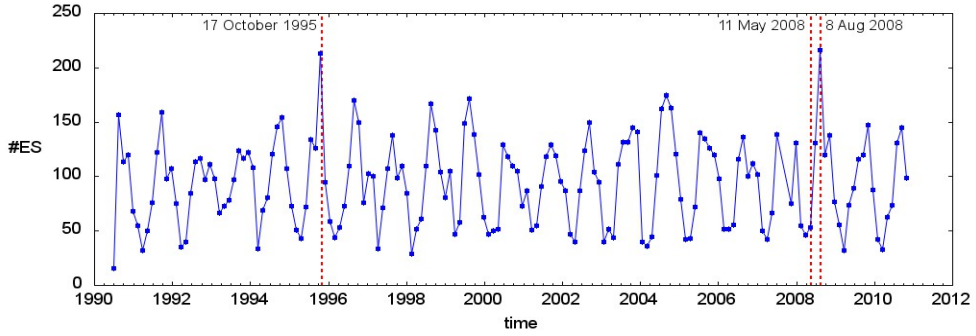


Figure 11: Distribution of the microscopic changes #ES discovered from the NOAA dataset as a function of time.

As for the NOAA domain (Figure 11), there are several macroscopic changes, those which have greater impact on the network are characterized by more than 150 microscopic changes. In particular, there are two macroscopic changes with highest number, they start on October 17th, 1995 and August 2008, 9th respectively. We deepened the microscopic changes corresponding to the two points and spotted several emerging subnetworks in common, two are listed in the following:

P_3: {(10.0_300.0, 10.0_305.0, bins_(70,80]__(80,90]),
(10.0_300.0, 7.5_297.5, bins_(80,90]__(90,+∞),
(12.5_297.5, 7.5_297.5, bins_(80,90]__(90,+∞))}

$GR_{\text{[1995 Feb 02, 1995 Nov 30], [1995 Feb 02, 1995 Oct 16]}}(P_3) = 1.39$
$macroChange = 0.66$
$GR_{\text{[2008 Jun 25, 2008 Sep 22], [2008 Jun 25, 2008 Aug 08]}}(P_3) = 1.27$
$macroChange = 0.51$
$minSUP = 0.5$, $minMC = 0.35$ and $minGR = 1$

P_4: {(10.0_295.0, 10.0_300.0,bins_(80,90]__(90,+∞)),
(10.0_300.0, 7.5_297.5,bins_(80,90]__(90,+∞)),
(7.5_297.5, 7.5_302.5,bins_(80,90]__(90,+∞))}

$GR_{\text{[1995 Feb 02, 1995 Nov 30], [1995 Feb 02, 1995 Oct 16]}}(P_4) = 1.35$
$macroChange = 0.66$
$GR_{\text{[2008 Jun 25, 2008 Sep 22], [2008 Jun 25, 2008 Aug 08]}}(P_4) = 1.24$
$macroChange = 0.51$
$minSUP = 0.5$, $minMC = 0.35$ and $minGR = 1$

P_3 and P_4 are both associated to two macroscopic changes of different quantities, they terminate on November 30th, 1995 and September 22th, 2008 respectively.

By mapping the nodes of P_3 and P_4 into a geodesic space, we see they identify two regions (three distant location per subnetwork), both cover approximately the area of the state of Venezuela and part of the Caribbean sea, where there are small differences in terms of relative humidity (the edge labels refer to consecutive ranges). This meteorological scenario becomes less frequent (the growth-rate decreases) over the window [1995 Oct 17, 1995 Nov 30] and [2008 Aug 09, 2008 Sep 22] respectively, which suggests the possibility of a different behavior, in the same geographic area, occurred before or after those two macroscopic changes. In fact, by inspecting the results, we find a macroscopic change between the windows [2008 May 11, 2008 Jun 25] and [2008 May 11, 2008 August 08] in which P_3 is absent (no relevant variation), but there is a new emerging subnetwork worthy of being analyzed:

P_5: {(10.0_300.0, 10.0_305.0,bins_(70,80]__(80,90]),
(10.0_300.0, .5_297.5,bins_(80,90]__(90,+∞])}

$GR_{[2008\ May\ 11,\ 2008\ Jun\ 25],\ [2008\ May\ 11,\ 2008\ Aug\ 8]}(P_5) = 2.35$
$macroChange = 0.62$
$minSUP = 0.5$, $minMC = 0.35$ and $minGR = 1$

P_5 is more general than P_3 (according to the generality order $\geqslant$, Section 3), but, contrarily to P_3, its frequency increases. Thus, we observe that the decrease of the relative humidity in the region of P_3 in the window [2008 Aug 8, 2008 Sep 22] is anticipated by the increase in the region P_5, which is smaller than that of P_3, in [2008 May 11, 2008 Aug 8].

6. Conclusions

In this paper we have investigated the problem of identifying relevant changes in evolving network data, where the changes can be distinguished in two categories: macroscopic changes and microscopic changes. Macroscopic changes have impact on a large portion of the network, whereas microscopic changes occur in specific portions of the networks. This analysis is motivated by typical real streaming network data, where we can observe *i)* abrupt changes, which can cause variations of the network both at the macroscopic and microscopic levels, and *ii)* gradual changes, which can cause only variations of the network at the microscopic level.

The system presented in the paper, called KARMA, is able to simultaneously extract macroscopic changes and microscopic changes by exploiting the fact that they are inevitably related each other. Algorithmically, KARMA is based on the concept of emerging subnetworks, that is, a kind of frequent patterns which represents connected subgraphs whose support changes over time. Contrary to many data stream mining algorithms for frequent pattern mining, which work on sliding windows of fixed size, our approach uses multiple windows to avoiding manual definition of the window size. This allows the system to truly adapt to the (network) data distribution over the stream.

We have evaluated KARMA on real-world network data streams having different properties and generated in different domains (social, technological and scientific). The experiments have mainly shown *i)* the efficiency of KARMA in comparison with competitors, *ii)* the validity of some discoveries in the domain of the datasets, *iii)* usefulness of the changes detected in the study of the domain under consideration.

For future work, we plan to investigate two main research directions: *i)* use solutions of big data analytics to detect changes in very large networks, *ii)* study the periodicity over time of macroscopic and microscopic changes.

Acknowledgments

The authors would like to acknowledge the support of the *i)* European Commission through the project "MAESTRA Learning from Massive, Incompletely annotated, and Structured Data" (Grant no. ICT-2013-612944) and *ii)* Apulia Regional Government through the project "Computer-mediated collaboration in creative projects" (8GPS5R0) collocated in "Intervento cofinanziato dal Fondo di Sviluppo e Coesione 2007-2013 – APQ Ricerca Regione Puglia -Programma regionale a sostegno della specializzazione intelligente e della sostenibilita' sociale ed ambientale - FutureInResearch".

References

[1] D. Kifer, Change detection on streams, in: L. Liu, M. T. Özsu (Eds.), Encyclopedia of Database Systems, Springer US, 2009, pp. 317–321. doi:10.1007/978-0-387-39940-9_49.

[2] L. I. Kuncheva, Change detection in streaming multivariate data using likelihood detectors, IEEE Trans. Knowl. Data Eng. 25 (2013) 1175–1180.

[3] F. Cao, J. Z. Huang, J. Liang, Trend analysis of categorical data streams with a concept change method, Inf. Sci. 276 (2014) 160–173.

[4] A. McGregor, Graph stream algorithms: A survey, SIGMOD Rec. 43 (2014) 9–20.

[5] F. S. F. Pereira, S. de Amo, J. Gama, On using temporal networks to analyze user preferences dynamics, in: T. Calders, M. Ceci, D. Malerba (Eds.), Discovery Science - 19th International Conference, DS 2016, Bari, Italy, October 19-21, 2016, Proceedings, volume 9956 of *Lecture Notes in Computer Science*, 2016, pp. 408–423. doi:10.1007/978-3-319-46307-0_26.

[6] J. Gama, M. M. Gaber, Learning from Data Streams: Processing Techniques in Sensor Networks, 1 ed., 2007.

[7] G. S. Manku, R. Motwani, Approximate frequency counts over data streams, PVLDB 5 (2012) 1699.
[8] C. Lin, D. Chiu, Y. Wu, A. L. P. Chen, Mining frequent itemsets from data streams with a time-sensitive sliding window, in: H. Kargupta, J. Srivastava, C. Kamath, A. Goodman (Eds.), Proceedings of the 2005 SIAM International Conference on Data Mining, SDM 2005, Newport Beach, CA, USA, April 21-23, 2005, SIAM, 2005, pp. 68–79. doi:10.1137/1.9781611972757.7.
[9] J. Leskovec, L. Backstrom, R. Kumar, A. Tomkins, Microscopic evolution of social networks, in: Y. Li, B. Liu, S. Sarawagi (Eds.), Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008, ACM, 2008, pp. 462–470. doi:10.1145/1401890.1401948.
[10] G. Widmer, M. Kubat, Effective learning in dynamic environments by explicit context tracking, in: P. B. Brazdil (Ed.), Machine Learning: ECML-93, Springer Berlin Heidelberg, Berlin, Heidelberg, 1993, pp. 227–243.
[11] S. J. Delany, P. Cunningham, A. Tsymbal, L. Coyle, A case-based technique for tracking concept drift in spam filtering, Knowledge-Based Systems 18 (2005) 187 – 195. AI-2004, Cambridge, England, 13th-15th December 2004.
[12] J. Sun, H. Fujita, P. Chen, H. Li, Dynamic financial distress prediction with concept drift based on time weighting combined with adaboost support vector machine ensemble, Knowledge-Based Systems 120 (2017) 4 – 14.
[13] R. Ahmed, G. Karypis, Algorithms for mining the evolution of conserved relational states in dynamic networks, Knowledge Information Systems 33 (2012) 603–630.
[14] M. Berlingerio, F. Bonchi, B. Bringmann, A. Gionis, Mining graph evolution rules, in: Proc. of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part I, ECML PKDD '09, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 115–130.
[15] E. Desmier, M. Plantevit, C. Robardet, J. Boulicaut, Granularity of co-evolution patterns in dynamic attributed graphs, in: H. Blockeel, M. van Leeuwen, V. Vinciotti (Eds.), Advances in Intelligent Data Analysis XIII - 13th International Symposium, IDA 2014, Leuven, Belgium, October 30 - November 1, 2014. Proceedings, volume 8819 of *Lecture Notes in Computer Science*, Springer, 2014, pp. 84–95. doi:10.1007/978-3-319-12571-8_8.
[16] Y. S. Koh, CD-TDS: change detection in transactional data streams for frequent pattern mining, in: 2016 International Joint Conference on Neural Networks, IJCNN 2016, Vancouver, BC, Canada, July 24-29, 2016, IEEE, 2016, pp. 1554–1561. URL: https://doi.org/10.1109/IJCNN.2016.7727383, doi:10.1109/IJCNN.2016.7727383.
[17] C. Loglisci, M. Ceci, D. Malerba, Discovering evolution chains in dynamic networks, in: NFMCP, 2012, pp. 185–199.
[18] C. Loglisci, M. Ceci, D. Malerba, Relational mining for discovering changes in evolving networks, Neurocomputing 150, Part A (2015) 265 – 288.
[19] P. N. E. Nohuddin, F. Coenen, R. Christley, C. Setzkorn, Y. Patel, S. Williams, Finding "interesting" trends in social networks using frequent pattern mining and self organizing maps, Knowl.-Based Syst. 29 (2012) 104–113.
[20] L. Akoglu, H. Tong, D. Koutra, Graph-based anomaly detection and description: A survey, CoRR abs/1404.4679 (2014).
[21] D. Pham, S. Venkatesh, M. Lazarescu, B. Saha, Anomaly detection in large-scale data stream networks, Data Min. Knowl. Discov. 28 (2014) 145–189.
[22] Y. Yang, J. X. Yu, H. Gao, J. Pei, J. Li, Mining most frequently changing component in evolving graphs, World Wide Web 17 (2014) 351–376.
[23] Z. Liu, J. X. Yu, Discovering burst areas in fast evolving graphs, in: H. Kitagawa, Y. Ishikawa, Q. Li, C. Watanabe (Eds.), Database Systems for Advanced Applications, 15th International Conference, DASFAA 2010, Tsukuba, Japan, April 1-4, 2010, Proceedings, Part I, volume 5981 of *Lecture Notes in Computer Science*, Springer, 2010, pp. 171–185. doi:10.1007/978-3-642-12026-8_15.
[24] Y. Zhang, H. Chen, J. Lu, G. Zhang, Detecting and predicting the topic change of knowledge-based systems: A topic-based bibliometric analysis from 1991 to 2016, Knowledge-Based Systems 133 (2017) 255 – 268.
[25] J. Gama, M. M. Gaber, Learning from data streams: processing techniques in sensor networks, Springer, 2007.
[26] P. Tan, M. Steinbach, V. Kumar, Introduction to Data Mining, Addison-Wesley, 2005.
[27] G. Dong, J. Li, Efficient mining of emerging patterns: Discovering trends and differences, in: U. M. Fayyad, S. Chaudhuri, D. Madigan (Eds.), Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, August 15-18, 1999, ACM, 1999, pp. 43–52. doi:10.1145/312129.312191.
[28] M. J. Zaki, S. Parthasarathy, M. Ogihara, W. Li, New algorithms for fast discovery of association rules, in: D. Heckerman, H. Mannila, D. Pregibon (Eds.), Proceedings of the Third International Conference on Knowledge Discovery and Data Mining (KDD-97), Newport Beach, California, USA, August 14-17, 1997, AAAI Press, 1997, pp. 283–286.
[29] R. Rymon, Search through systematic set enumeration, in: B. Nebel, C. Rich, W. R. Swartout (Eds.), Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR'92). Cambridge, MA, October 25-29, 1992., Morgan Kaufmann, 1992, pp. 539–550.
[30] F. Coenen, G. Goulbourne, P. Leng, Tree structures for mining association rules, Data Mining and Knowledge Discovery 8 (2004) 25–51.
[31] M. Kuramochi, G. Karypis, An efficient algorithm for discovering frequent subgraphs, IEEE Trans. Knowl. Data Eng. 16 (2004) 1038–1051.
[32] C. Jiang, F. Coenen, M. Zito, A survey of frequent subgraph mining algorithms, Knowledge Eng. Review 28 (2013) 75–105.
[33] M. Quadrana, A. Bifet, R. Gavaldà, An efficient closed frequent itemset miner for the MOA stream mining system, AI Commun. 28 (2015) 143–158.
[34] L. Lhote, F. Rioult, A. Soulet, Average number of frequent (closed) patterns in bernouilli and markovian databases, in: Proceedings of the 5th IEEE International Conference on Data Mining (ICDM 2005), 27-30 November 2005, Houston, Texas, USA, 2005, pp. 713–716. URL: https://doi.org/10.1109/ICDM.2005.31, doi:10.1109/ICDM.2005.31.
[35] X. Yan, J. Han, gspan: Graph-based substructure pattern mining, in: Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002), 9-12 December 2002, Maebashi City, Japan, IEEE Computer Society, 2002, pp. 721–724. doi:10.1109/ICDM.2002.1184038.
[36] Y. Chi, H. Wang, P. S. Yu, R. R. Muntz, Catch the moment: maintaining closed frequent itemsets over a data stream sliding window, Knowl. Inf. Syst. 10 (2006) 265–294.
[37] P. A. Schrodt, S. G. Davis, J. L. Weddle, Political Science: KEDS–A Program for the Machine Coding of Event Data, Social Science Computer Review 12 (1994) 561.

[38] U. Brandes, J. Lerner, Visualization of conflict networks, Nato Security Through Science Series - E: Human and Societal Dynamics 36 (2008) 169.

[39] S. Bell, A. McDiarmid, J. Irvine, Nodobo: Mobile phone as a software sensor for social network research, in: Proceedings of the 73rd IEEE Vehicular Technology Conference, VTC Spring 2011, 15-18 May 2011, Budapest, Hungary, IEEE, 2011, pp. 1–5. doi:10.1109/VETECS.2011.5956319.

[40] E. Kalnay, M. Kanamitsu, R. Kistler, W. Collins, D. Deaven, L. Gandin, M. Iredell, S. Saha, G. White, J. Woollen, Y. Zhu, A. Leetmaa, B. Reynolds, M. Chelliah, W. Ebisuzaki, W. Higgins, J. Janowiak, K. C. Mo, C. Ropelewski, J. Wang, R. Jenne, D. Joseph, The NCEP/NCAR 40-Year Reanalysis Project., Bulletin of the American Meteorological Society 77 (1996) 437–472.

[41] G. B. Tran, M. Alrifai, D. Q. Nguyen, Predicting relevant news events for timeline summaries, in: L. Carr, A. H. F. Laender, B. F. Lóscio, I. King, M. Fontoura, D. Vrandecic, L. Aroyo, J. P. M. de Oliveira, F. Lima, E. Wilde (Eds.), 22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013, Companion Volume, International World Wide Web Conferences Steering Committee / ACM, 2013, pp. 91–92.

[42] X. Yan, J. Han, gspan: Graph-based substructure pattern mining, in: Technical Report UIUCDCS-R-2002-2296, Department of Computer Science, University of Illinois at Urbana-Champaign, 2002.

[43] J. Han, M. Kamber, Data Mining: Concepts and Techniques, Morgan Kaufmann, 2002, pp. 535–589.

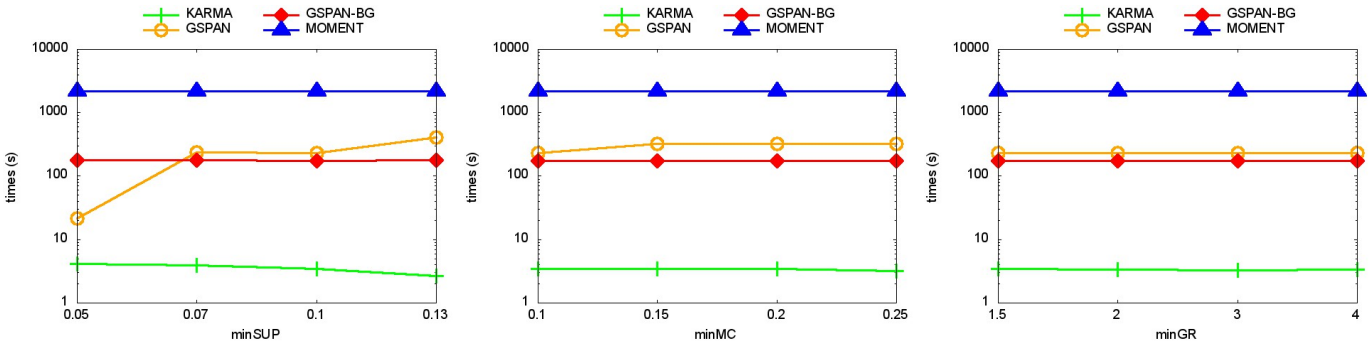
Appendix A. Evaluation of the variants

In this section, we evaluate three variants of KARMA obtained by replacing the algorithm of discovery of subnetworks described in Section 4.1 with the algorithms gSpan and MOMENT.

The algorithm gSpan works on frequent subgraphs, but does not operate in the streaming scenario. It mines frequent subgraphs by combining the pattern-growth blueprint with the depth-first approach. We use two alternative gSpan versions. The first implementation is an extension of the native algorithm in that it mines input graphs which have edges with multiple labels [42, 43], but it discovers subgraphs which have edges with only one label[5]. We will refer to the experimental results of this version as "GSPAN". In the second version, to overcome the limitation on the single labeled edges, we use a different representation based on bipartite graphs. More precisely, one partition of the graphs is built with the node labels, the other one with the edge labels. We will refer to the experimental results of this version as "GSPAN-BG".

The algorithm MOMENT works in the streaming scenario, does not analyze network data but itemset-based data streams and discovers closed itemsets frequent in a window sliding over the stream. The input set of the items is built with the sets of triples of the network snapshots.

For these comparisons we chose the dataset KEDS because it has the longest time-interval. The variants have been tested under the perspectives of the running times, distance between consecutive macroscopic changes (that is, the average number of windows) and total number of microscopic changes (sum of #ES with #JES). While the number of discovered microscopic changes refers to the topological regularities which exhibit changes, the distance between consecutive macroscopic changes indicates the number of windows in which macroscopic changes are identified. Experiments were performed by tuning the thresholds minSUP, minMC and minGR. The initial width of W' is set to 365. The results are reported in Figures A.1, A.2, A.3.

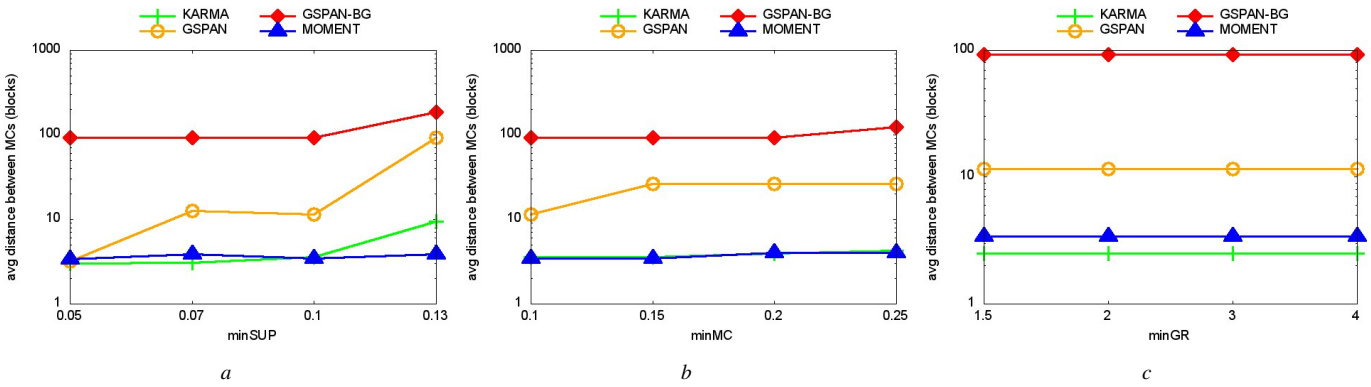


a *b* *c*

Figure A.1: Running times of three different algorithms for mining subnetworks when tuning the thresholds $minSUP$ (a), $minMC$ (b) and $minGR$ (c). The settings are $minGR$=2, $minMC$=0.1 for (a); $minSUP$=0.1, $minGR$=2 for (b); $minSUP$=0.1, $minMC$=0.1 for (c).

In Figure A.1 (data are plotted in log scale), we can see the running times of KARMA are significantly lower than those of GSPAN, GSPAN-BG and MOMENT. As for GSPAN and GSPAN-BG, we can motivate this difference with two factors, *i)* the use of a canonical labeling system, which does not exclude the generation of subgraphs that might have already been created, and *ii)* the mechanism of the computation of the support, which may require the identification of subgraph isomorphism for exact counting (which is not a required property for the task we consider). As for MOMENT, *times* remains unchanged since we set the maximum running time for the experiments to 6 hours of uninterrupted execution. This result is motivated by the mechanism of updating the support of MOMENT. Indeed, it scans the incoming transactions singularly, while our approach exploits querying techniques to determine the support in only one operation.

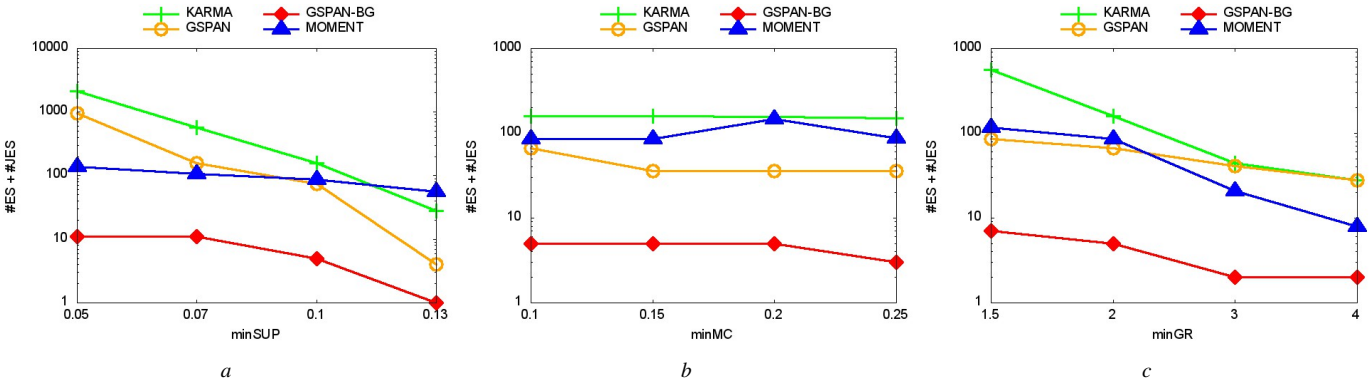
[5]The implementation of the gSpan algorithm to mine graphs with multi-labelled edges is available at the web site https://www.cs.ucsb.edu/x̃yan/software/gSpan.htm



a *b* *c*

Figure A.2: Average size of windows obtained between macroscopic changes when tuning the thresholds $minSUP$ (a), $minMC$ (b) and $minGR$ (c). The settings are $minGR$=2, $minMC$=0.1 for (a); $minSUP$=0.1, $minGR$=2 for (b); $minSUP$=0.1, $minMC$=0.1 for (c).

Figure A.2 reports the average distance between two macroscopic changes. We can see that the values of KARMA are lower than GSPAN, GSPAN-BG and MOMENT, meaning that our approach is able to detect macroscopic changes more frequently, and therefore it is more sensitive to the variations of the lattice than the other three versions. As expected, this result is strictly related to the results of microscopic changes (Figure A.3). Indeed, the difficulty to detect changes in the lattice (especially for GSPAN and GSPAN-BG) augments the complexity to identify variations in the sets of (frequent and infrequent) subnetworks. This leads to a lower number of discovered microscopic changes, which we attribute to the difficulty to capture the variations of the network over the stream (MOMENT), or to the difficulty to generate subnetworks to model the whole network (GSPAN and GSPAN-BG).



a *b* *c*

Figure A.3: Total number of microscopic changes (#ES + #JES) discovered by the variants when tuning the thresholds $minSUP$ (a), $minMC$ (b) and $minGR$ (c). The settings are $minGR$=2, $minMC$=0.1 for (a); $minSUP$=0.1, $minGR$=2 for (b); $minSUP$=0.1, $minMC$=0.1 for (c).

Figure A.3 (plotted in log scale) reports the number of the emerging subnetworks (#ES+#JES). We can see that KARMA discovers a larger set of microscopic changes as compared to the other three variants. As for GSPAN, the main reason is that the generated subgraphs cannot have pairs of the same nodes connected by different edges, which reduces the number of potential subnetworks. The explanation for the results of GSPAN-BG can be found in the bipartite representation, which, although allowing to model multiple labeled edges, does not facilitate the discovery of microscopic changes. This is because many subnetworks are disconnected (e.g., some edges do not connect two nodes but only one) and thus they are not considered for the computation of the statistics. As for MOMENT, we ascribe its behavior to the characteristics of the data stream on which that variant works. Indeed, using a large set of items to represent a streaming network leads inevitably to producing a stream of very sparse vectors, which tends to lower the support of the patterns and flatten their variations.