Constraint-based Pattern Mining in Dynamic Graphs

Céline Robardet
Université de Lyon, INSA-Lyon, CNRS, LIRIS UMR 5205
F-69621 Villeurbanne, France
Email: Celine.Robardet@insa-lyon.fr

***Abstract*—Dynamic graphs are used to represent relationships between entities that evolve over time. Meaningful patterns in such structured data must capture strong interactions and their evolution over time. In social networks, such patterns can be seen as dynamic community structures, i.e., sets of individuals who strongly and repeatedly interact. In this paper, we propose a constraint-based mining approach to uncover evolving patterns. We propose to mine dense and isolated subgraphs defined by two user-parameterized constraints. The temporal evolution of such patterns is captured by associating a temporal event type to each identified subgraph. We consider five basic temporal events: The formation, dissolution, growth, diminution and stability of subgraphs from one time stamp to the next. We propose an algorithm that finds such subgraphs in a time series of graphs processed incrementally. The extraction is feasible due to efficient patterns and data pruning strategies. We demonstrate the applicability of our method on several real-world dynamic graphs and extract meaningful evolving communities.**

***Keywords*-dynamic graph; local pattern; evolving pattern**

I. Introduction

Graphs are data models used to represent any kind of relationship among various entities. Graphs are most widely used for technological, sociological and scientific applications. For instance, one can analyze innovation dissemination, information diffusion and epidemiology. The study of such graphs has attracted much attention in the last few years and has proceeded along two main tracks: (a) The analysis of graph properties, such as degree distribution and diameter [1], and (b) the definition and the extraction of more sophisticated properties using the pattern mining framework [2], [3], [4], [5]. To probe relationships in real-world systems, it seems more adequate to look for temporal interactions, since most of the previously mentioned graphs tend to change dynamically. If these tools are adapted to static graphs, it seems more adequate, when analyzing dynamic graphs, to look for temporal interactions. As new vertices and edges appear while others disappear over time, it seems decisive to examine deeply the evolution of such dynamic graphs. Furthermore, there is a crucial need for incremental methods that enable to find groups of associated vertices and detect how these structures change over time.

Interesting subgraph patterns are often assumed to be highly connected within the subgraph [4] and isolated from the rest of the graph, *i.e.* having few links with the remaining vertices of the graph. These properties can be captured by measures such as modularity [6], used to find disjoint communities forming a partition. The modularity of a partition of vertices is the number of edges inside the clusters (as opposed to crossing between clusters), minus the expected number of such edges if the graph was random conditioned on its degree distribution. However, this measure has two main drawbacks: (a) It suffers from an intrinsic resolution scale that prevents it from detecting small communities and favors clusters of similar size [7]; (b) Maximizing the modularity is NP-complete [8].

Instead of directly looking for a global structure of the graph, such as a partition of the vertices, it can be more efficient to proceed in two steps. One might first compute subgraphs that capture locally strong associations between vertices and then use these local patterns to construct a global model of the graph's dynamics. Such a framework provides more interesting patterns when the analyst can specify his inclination by means of constraints. Local patterns are characterized by the fact that their validity can be evaluated independently from other patterns [9]. Many pattern-mining-under-local-constraints techniques have been studied extensively during the last decade, highlighting the crucial role of local constraint properties in tractability issues. This process results in a possibly large and unstructured set of patterns. To turn this result into a true nugget of knowledge that can be readily interpreted, these local patterns have to be post-processed to form global models (e.g., classifiers, clusterings), i.e., sets of patterns satisfying global constraints. Such a mining task is called a Local-to-Global approach.

Fully connected subgraphs, also called cliques, are local patterns that have been used to construct communities. For example, Palla et al. [10] put all the cliques of size k that share $k-1$ vertices together. Such structures can be explored systematically with a deterministic algorithm. Although cliques are a popular way of capturing dense subgraphs, it often fails with experimental data because of the high level of noise. In such noisy data, some links may be missing even in dense substructures. To cope with this problem, a relaxed definition of cliques has been proposed: Pseudo-cliques are connected subgraphs with a density (proportion of connected peer vertices on the total number of peers) higher than a given threshold. Recent research results show that the constraint defining pseudo-cliques can be efficiently used in a mining algorithm [5]. In this paper, we extend this result and derive a new algorithm that extracts isolated pseudo-cliques and their evolution in time. We consider five basic temporal event types: The formation, dissolution, growth, diminution and stability of pseudo-cliques. Such evolving patterns allow us to describe the processes by which communities come together, attract new members, and develop over time. The use of complete solvers allows us to answer user constrained queries without uncertainty.

The article is organized as follows. The next section is dedicated to related work, section 3 presents the constraints that define the pseudo-cliques that are to be extracted in a static graph and the algorithm that handles these constraints. Section 4 introduces the evolving pattern types and the algorithm EVOLVING-SUBGRAPHS that mines them. Section 5 reports the results of our experimental evaluations and the last section presents some conclusions and future work.

II. RELATED WORK

In the past few years, many constraint-based graph mining approaches have been proposed. Zhu et al. [2] study the pruning properties of graph structural constraints such as density and diameter. They propose a general mining framework that allows pruning on both patterns and data spaces. Mining local patterns specific to graphs have also been considered. Classical graph properties like cliques or idependent sets can uncover new interesting information in the data. Efficient computation of maximal cliques is achieved in [11]. Several papers propose to relax the clique property by allowing some links to be missing. Hämäläinen et al. [3] define strongly self-referring subgraphs as a set of nodes S whose vertices are connected to at least σ nodes of S. Another way of relaxing this constraint is to use pseudo-cliques, defined as subgraphs having a density greater than a user-defined threshold. It was first studied by [4], but the complete exploitation of the loose anti-monotonicity property of the pseudo-clique constraint was only accomplished in [5] where a polynomial delay algorithm that extracts all pseudo-cliques is proposed.

There is an increasing interest in mining dynamic graphs. Borgwardt et al. [12] apply frequent-subgraph mining algorithms to time series of graphs and extract subgraphs that are frequent within the set of graphs. The extraction of periodic or near periodic subgraphs was considered in [13] where the problem is shown to be polynomial. In this paper, we are interested in finding evolving patterns in dynamic networks. Our approach starts by the extraction of local patterns in static graphs, and then combines the obtained patterns to construct a global model [9] of the graph dynamic.

III. CONSTRAINED SUBGRAPHS IN STATIC GRAPHS

Let us first present the static pattern type we are interested in. Let $G = (V, E)$ be a simple undirected graph with a vertex set V and an edge set $E \subseteq V \times V$. The degree $\deg(u)$ of a vertex u is the number of vertices v adjacent to u, i.e., $\deg(u) = |\{v \in V \mid \{u, v\} \in E\}|$. The subgraph induced by a subset of vertices S ($S \subseteq V$) is the graph $G_S = (S, E_S)$ where $E_S = \{\{u, v\} \mid \{u, v\} \in E \wedge u, v \in S\}$.

Subgraphs of interest are usually those made of vertices that have a high density of edges. The density of a subgraph is defined as the number of edges in the subgraph divided by the maximal number of possible edges. A clique is a subgraph with a density of 1. To relax this strong property, we can consider subgraphs of a density at least equal to a user-defined threshold. Such subgraphs are usually called pseudo-cliques or quasi-cliques.

Definition 1 (Pseudo-clique): Given a user-defined threshold $\sigma \in [0, 1]$ and a set of vertices $S \subseteq V$ of size n, the subgraph $G_S = (S, E_S)$ induced by S is a pseudo-clique iff it is connected and $\frac{2|E_S|}{n(n-1)} \geq \sigma$. Let us denote $\deg_S(u) = |\{v \mid \{u, v\} \in E_S\}|$, the constraint can then be rewritten as follows

$$\mathcal{P}_{pc}(S, \sigma) \equiv \frac{\sum_{u \in S} \deg_S(u)}{n(n-1)} \geq \sigma \tag{1}$$

Constraint-based mining algorithms require to take advantage of the constraints to prune huge parts of the search space which can not contain valid patterns. Pruning based on monotonic or anti-monotonic constraints has been proved efficient on hard problems: When a candidate does not satisfy the constraint then neither of its generalizations or specializations can satisfy it. Let us first remark that pseudo-clique constraint is not anti-monotonic with respect to the enumeration of subgraphs based on the set inclusion of their vertex sets: Expanding a set S of n vertices could make the density $\left(\frac{2|E_S|}{n(n-1)}\right)$ increase or decrease. However, this constraint is loose anti-monotonic, that is to say, pseudo-cliques can always be grown from a smaller pseudo-clique with one vertex less [2].

Property 1 (Loose anti-monotonicity of pseudo-clique): Pseudo-clique constraint is loose anti-monotonic [14], i.e., $\mathcal{P}_{pc}(S, \sigma) \Rightarrow \exists v \in S \; such \; that \; \mathcal{P}_{pc}(S \setminus \{v\}, \sigma)$.

Proof: Suppose the subgraph S satisfies $\mathcal{P}_{pc}(S, \sigma)$. Let $v^\star$ be a vertex of S having the smallest degree on S ($\deg_S(v^\star) = \min_{u \in S} \deg_S(u)$), n be the size of S and $S^\star = S \setminus \{v^\star\}$. Then, we have

$$\begin{aligned} \sum_{u \in S} \deg_S(u) &= \sum_{u \in S^\star} \deg_S(u) + \deg_S(v^\star) \\ &= \sum_{u \in S^\star} \deg_{S^\star}(u) + 2\deg_S(v^\star) \geq \sigma n(n-1) \end{aligned}$$

- If $\deg_S(v^\star) \leq \sigma(n-1)$, then $\sum_{u \in S^\star} \deg_{S^\star}(u) \geq \sigma n(n-1) - 2\sigma(n-1) \geq \sigma(n-1)(n-2)$ and $\mathcal{P}_{pc}(S^\star, \sigma)$ is satisfied.
- Otherwise, $\forall u \in S$, $\deg_S(u) > \sigma(n-1)$, and we have $\sum_{u \in S^\star} \deg_{S^\star}(u) = \sum_{u \in S} \deg_S(u) - 2\deg_S(v^\star) \geq (n-2)\deg_S(v^\star) \geq (n-2)\sigma(n-1)$ and $\mathcal{P}_{pc}(S^\star, \sigma)$ is also satisfied. ■

To be efficient, the enumeration process must tap the pruning power from the loose anti-monotonicity of pseudo-cliques. It is clear, from the proof of Property 1, that adding the vertex $v \in V$ that satisfies Equation (2) to a current pseudo-clique S, leads to a pseudo-clique unless none of the supersets of S is a pseudo-clique.

$$\deg_{S\cup\{v\}}(v) = \min_{u\in S\cup\{v\}} \deg_{S\cup\{v\}}(u) \qquad (2)$$

Thus, an efficient algorithm enumerates vertices recursively by finding at each iteration the vertex[1] v that satisfies Equation (2) and stop the enumeration if $\mathcal{P}_{pc}(S \cup \{v\}, \sigma)$ is not satisfied. This leads to a polynomial delay time algorithm, that is to say the time needed to generate each single pseudo-clique is bounded by a polynomial in the size of the input graph. The efficiency of the algorithm relies on the loose anti-monotonic property of the pseudo-clique constraint that guarantees that each pseudo-clique is generated at most once and that no invalid pseudo-cliques are generated.

Uno proposes in [5] a method to solve Equations (1) (on $S \cup \{v\}$) and (2) efficiently. As $\sum_{u\in S\cup\{v\}} \deg_S(u) = \sum_{u\in S} \deg_S(u) + 2\deg_S(v)$, $\mathcal{P}_{pc}(S\cup\{v\},\sigma)$ is satisfied iff $\deg_S(v) \geq \frac{\sigma|S|(|S|+1)-\sum_{u\in S}\deg_S(u)}{2}$. Equation (1) is thus checked in constant time if $\sum_{u\in S}\deg_S(v)$ is stored and updated during the enumeration process. Equation (2) can trivially be checked in $(O(|V|)$, but Uno shows in [5] how to do it in time $O(\deg_S(v))$.

Pseudo-cliques are local patterns of interest to capture strong (but not necessarily perfect) associations in a graph. But, not all the pseudo-cliques of a graph are of importance: Some of them have many links to external vertices and others are redundant. To identify the most useful pseudo-cliques, we consider two other constraints that coerce the patterns to be isolated and maximal. The isolation constraint (see Definition 2) imposes a maximum to the average number of external links per vertex.

Definition 2 (Isolated constraint): Given a user defined threshold $\gamma \in \mathbb{R}$, a subgraph S is isolated iff

$$\mathcal{P}_i(S,\gamma) \equiv \frac{\sum_{u\in S}(\deg(u) - \deg_S(u))}{|S|} \leq \gamma.$$

$\mathcal{P}_i(S,\gamma)$ is loose anti-monotonic (proof is omitted due to space constraint).

However, the combination of two loose anti-monotonic constraints (Definitions 1 and 2) is not necessarily a loose anti-monotonic constraint. To enumerate isolated pseudo-cliques in a single process, the algorithm should find the vertex v that satisfies both Equation (2) and $v = \arg\max_{u\in S\cup\{v\}} \Big(\deg(u) - \deg_{S\cup\{v\}}(u)\Big)$. The conjunction of these equations characterizes the vertex leading to an isolated pseudo-clique. Such a vertex does not necessarily exist and thus $\mathcal{P}_{pc} \wedge \mathcal{P}_i$ is not loose anti-monotonic. The two constraints cannot be ensured at the same time by an algorithm that uses both loose anti-monotonic constraints. Hence, we propose to use $\mathcal{P}_i$ in a post-processing of the previously computed pseudo-cliques.

Extracting maximal patterns is even more difficult, since this constraint is global and requires to enumerate supersets of each candidate to check whether it is maximal. A practical approach consists in extracting locally maximal isolated pseudo-cliques as defined below.

Definition 3 (Pattern locally maximal): A subgraph S of size n is a local maximal isolated pseudo-clique if $\mathcal{P}_{pc}(S,\sigma) \wedge \mathcal{P}_i(S,\gamma)$ is satisfied and no superset of S of size $n+1$ satisfies these properties. This property is denoted $\mathcal{P}_{max-ipc}(S,\sigma,\gamma)$ and will henceforth be referred to as valid pseudo-cliques.

Thanks to this locally maximal constraint, the very large majority of non-maximal isolated pseudo-cliques are removed, while the time complexity of the extraction remains unchanged.

IV. Mining evolving subgraphs

The method explained in the previous section to compute valid pseudo-cliques gives unstructured and numerous patterns. These results are hence difficult (if not impossible) to interpret [9]. We propose to complement this first step, during which valid pseudo-cliques in static graphs are mined, with a second step that constructs a global model of the dynamic graph. We consider a dynamic graph $\mathcal{G} = (G^1, \dots, G^T)$ which is a time-series of graphs, where $G^t = (V^t, E^t)$ is the graph of vertices V^t and edges E^t observed at time t.

The typical questions we want to consider are:

- Do the strong interactions observed at time t grow, diminish or remain stable over time?
- When do the changes occur?

The objective here is to identify the temporal relationships that may occur between valid pseudo-cliques. We denote by $\mathcal{C}^t$ the set of subgraphs of G^t that satisfy $\mathcal{P}_{max-ipc}$. We consider five basic temporal relationships between couples of subgraphs from consecutive time stamps:

Stability: S is said to be stable at time t if it is a valid pseudo-clique at both times t and $t-1$: $S \in \mathcal{C}^t$ and $S \in \mathcal{C}^{t-1}$.

Growth: A subgraph S enlarges at time t if S is a valid pseudo-clique at time t and if a subpart of S was a valid pseudo-clique at time $t-1$: $S \in \mathcal{C}^t$ and $\exists R,\ R \subset S\ such\ that\ R \in \mathcal{C}^{t-1}$.

Diminution: A subgraph S shrinks at time t if S is a valid pseudo-clique at time t and if it is a subpart of a larger valid pseudo-clique of time $t-1$: $S \in \mathcal{C}^t$ and $\exists R,\ S \subset R\ such\ that\ R \in \mathcal{C}^{t-1}$.

[1]Note that if several vertices satisfy Equation (2), the one of smallest index is taken.

Extinction: A subgraph S disappears at time t if it was a valid pseudo-clique at time $t-1$ and if it is not involved in any previously defined temporal relationship at time t: $S \in \mathcal{C}^{t-1}$ and $\forall R, R \subseteq S, R \notin \mathcal{C}^t$ and $\forall R, S \subseteq R, R \notin \mathcal{C}^t$.
Emergence: A subgraph S emerges at time t if it is a valid pseudo-clique in G^t and if none of its subsets or supersets are valid pseudo-cliques in G^{t-1}: $S \in \mathcal{C}^t$ and $\forall R,\ R \subseteq S,\ R \notin \mathcal{C}^{t-1}$ and $\forall R,\ S \subseteq R,\ R \notin \mathcal{C}^{t-1}$.

Those temporal relationships correspond to global constraints used to identify the dynamics of strong associations in graphs. We now present an incremental algorithm that processes each static graph sequentially. Inspired by the Trie-based Apriori implementation [15], we propose to use a trie data structure (prefix tree) to store valid pseudo-cliques. Indeed, finding evolving patterns requires the evaluation of subset queries over valid pseudo-cliques of G^{t-1} and G^t. Such queries are computationally consuming and require special attention. Trie is appropriate for storing and retrieving any finite set. We use it here to retrieve the vertex sets defining valid pseudo-cliques.

Suppose that pseudo-cliques of $\mathcal{C}^{t-1}$ are stored in a trie $\mathcal{T}$. Each node of $\mathcal{T}$ consists of the set S of all the vertices of the pseudo-clique, a list of temporal states, a list of pointers to other trie nodes and a list of time stamps. When a new valid pseudo-clique of G^t is computed, its vertex set S is inserted in $\mathcal{T}$ recursively. Starting from the root node, we first go to the child corresponding to the first vertex of S and process the remainder of S recursively for that child. The recursion stops on a node whose vertex set is either S, or a prefix of S:

- In the first case, the temporal label *"Stability"* is pushed back in the temporal label list of the node and its time stamp is set to t.
- In the latter case, the node gets a new son with vertex set S, time stamp t and temporal label *"Emergent"*. Then we look whether S is involved in a growing evolving pattern. To do so, we have to retrieve all the subsets of S from $\mathcal{T}$ by means of the following doubly recursive procedure: We first go to the child corresponding to the first vertex of S and process the remainder of S recursively for that child and second discard the first vertex of S and process it recursively for the node itself. If there exists subsets of S that belongs to $\mathcal{T}$ with time stamp label $t-1$, then the temporal state associated to S is changed into *"Growth"* and pointers to the corresponding subsets are stored in the list associated to the node. Those nodes are also tagged to avoid their consideration in the following step.

Now that *"Stability"* and *"Growth"* patterns have been dealt with, we need to check whether the remaining nodes (those associated to pseudo-cliques of $\mathcal{C}^{t-1}$) have shrunk (*"Diminution"*) or completely disappeared (*"Extinction"*). As tries are more effective to find subsets than to find supersets, a second traversal of the trie is performed when all pseudo cliques of $\mathcal{C}^t$ have been processed. For all the nodes with time stamp $t-1$ that are not involved in a *"Stability"* or *"Growth"* pattern, the function that searches subsets is triggered. If there exists a subset that belongs to $\mathcal{C}^t$, the state of the first node is set to *"Diminution"* and pointers to the corresponding subsets are stored in the node list, otherwise the state is set to *"Extinction"*, the pattern is output and the node is removed from the trie.

V. Experimental results

We evaluate the added-value of our method EVOLVING-SUBGRAPHS and the general characteristics of evolving patterns subgraphs on three real-world dynamic networks: two dynamic sensor networks, IMOTE and MIT, and a dynamic mobility network Vélo'v, the shared bicycle system of Lyon. The main characteristics of these datasets are presented on Table I. All experiments were done on a Pentium 3 with 2 gigabytes of memory running on Linux.

Dataset	♯ Edges	♯ Timesteps	Avg. density
IMOTE	11785	282	0.025
MIT	107770	11763	0.001
Vélo'v	279208	930	0.003

Table I
DATASET CHARACTERISTICS

A. Dynamic sensor networks

Both studied mobility networks are based on sensor measurements. The IMOTE [16] data set has been collected during the Infocom 2005 conference. Bluetooth sensors were distributed to a set of participants who were asked to keep the sensors with them continuously. These sensors were able to detect and record the presence of other Bluetooth devices inside their radio-range neighborhood. The available data concerns 41 sensors over a period of 3 days. The MIT or *Reality Mining* [17] experimental data set is constituted of records from Bluetooth contacts for a group of cell-phones distributed to 100 MIT students during 9 months. Each cellular phone recorded the identities of all the devices present in its neighborhood. Note that the sensors have no localization capability, therefore we do not have information on the actual movements of individuals carrying the sensors. Since it may happen that a given sensor misses another one, we admit the existence of an undirected link between the two sensors as soon as one sensor sees the other. We study the IMOTE dataset over a typical day and the MIT data over a typical week. The number of edges of those graphs are reported in Figure 1. Both IMOTE and MIT graphs are sparse (the number of edges is low) and the number of edges exhibits large variations over time.

To densify the graphs and cope with the flickering edge problem that may occur with experimental data, we aggregate the graphs over a period of 15 minutes for IMOTE and 1 hour for MIT: in both dynamic graphs, an edge exists if

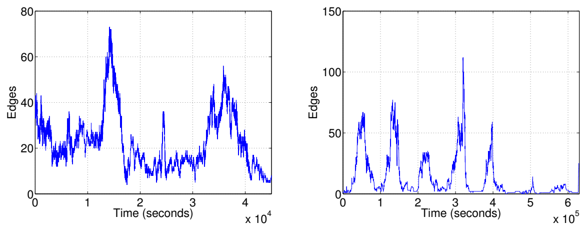

Figure 1. Number of edges, displayed as a function of time (IMOTE on the left and MIT on the right).

it appears at least once during the considered period. The resulting dynamic graphs have a maximum degrees of 25 for IMOTE and 22 for MIT.

We extract evolving subgraphs with several density values σ, the average number of out-of-subgraph edges per vertex being set to 4.5 ($\gamma = 4.5$) and the minimal size of the extracted valid pseudo-cliques set to 4 for IMOTE and to 3 for MIT. The total runtimes and number of computed patterns are shown on Figure 2 (left). These figures show that EVOLVING-SUBGRAPHS is tractable in terms of execution time since it succeeds to extract the patterns in less than 20 minutes for different σ values varying between 1 and 0.6. The computational time is proportional to the number of output patterns; that was expected according to the theoretical study of the time complexity of the pseudo-clique mining algorithm discussed in Section 3. The time required to compute evolving patterns generally decreases with σ as well as the number of extracted patterns.

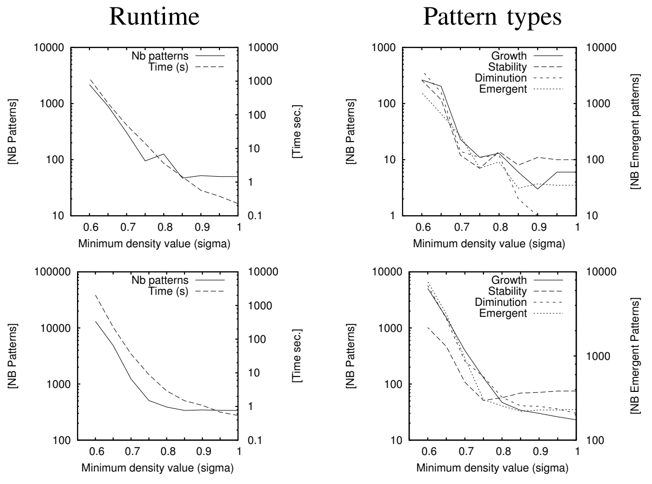

Figure 2. Runtime and number of extracted patterns (logarithmic scales) (left) and number of patterns of each type (right) for IMOTE (top) and MIT (bottom) dynamic graphs for different density threshold σ.

The numbers of evolving patterns of each type are shown on Figure 2 (right). As the number of *"Emergent"* patterns scales differently from other pattern types, their quantity is shown on the right ordinate axe, whereas the number of *"Growth"*, *"Stability"* and *"Diminution"* patterns are plotted using the left ordinate axe. Even though the number of patterns decreases with the density threshold, we can observe that the number of each type of patterns varies differently from one another.

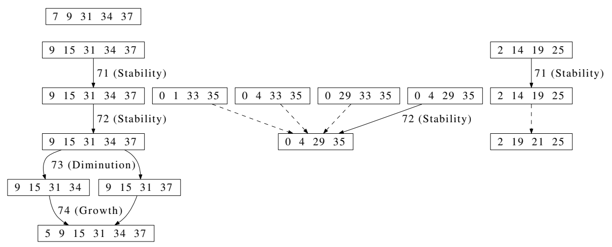

Figure 3. Display of the evolving patterns for IMOTE with $\sigma = 0.8$, $\gamma = 3$ and the minimum subgraph size equals 4 that occur in the morning.

Figure 3 shows the output of our method: nodes represent valid pseudo-cliques and the numbers they contain are vertices identifiers, solid arrows show evolving patterns and dashed arrows are drawn between following subgraphs that intersect. We can identify three main groups of people. The first one is composed of individuals 9, 15, 31, 34 and 37. This group appears at time stamp 71, splits around time stamp 73 into two groups that then merge and integrate an additional vertex 5. The second group is made up of individuals 0, 4, 29 and 35. Individuals 1 and 33 are nearby. This group is stable since it remains unchanged during two consecutive time stamps. The third group contains individuals 2, 14, 19 and 25 and is also stable.

B. Shared bicycle system Vélo'v

We analyze Lyon's shared bicycle system *Vélo'v* on the basis of the data provided by JCDecaux, promotor and operator of the program. The dataset contains all the bicycle trips that occurred between the 25th of May 2005 and the 12th of December 2007. During this period, there were more than 13 million hired bicycle trips. Each record is anonymized and presents the date and time as well as the station's ID for both the beginning and the end of the trip (their geographical location being known).

To analyse the *Vélo'v* dataset, we first aggregate the number of rentals for every days of the week and every hours over the two and a half years period of observation. We thus obtain 168 time stamps. Then, to leverage the most important links, we remove the edges that had less than 50 rentals over this period.

We compute the total number of extracted evolving patterns and EVOLVING-SUBGRAPHS runtime for several σ values, γ being set to 5 and the minimum subgraph size to 3. The results are similar to those on Figure 2 but are not shown due to space constraint.

Figure 4 displays the output of EVOLVING-SUBGRAPHS for the *Vélo'v* data set for time stamps between Monday 6

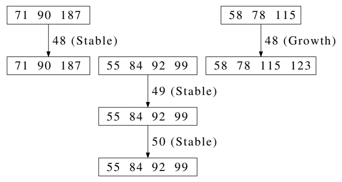

Figure 4. Display of the evolving patterns for *Vélo'v* with $\sigma = 0.9, \gamma = 5$ and the minimum subgraph size equals 3.

PM and Tuesday 7 AM. The analysis of this outcome carries interesting pieces of information: For example, at time stamp 48 (Tuesday 0 AM to 1 AM) the identified patterns give rise to the group of stations 58, 78, 115 that are located on the largest campus of Lyon. This pattern grows between 1 AM and 2 AM with the addition of the neighboring station 123. At the same time another subgraph appears, that contains stations 187, 71 and 90, that surround the main park of Lyon. Another important group is the one made of stations 55, 84, 92 and 99 that are all located in the 7th district of the city where the second largest campus is located. Those stations are located on the map of Lyon on Figure 5.

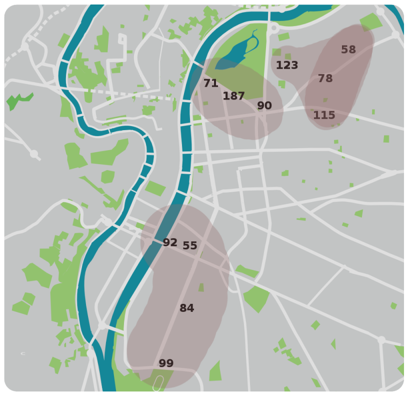

Figure 5. Location of *Vélo'v* stations in Lyon. Patterns from Figure 4 are shadowed on the map.

VI. Conclusion

In this paper, we considered the evolving-pattern mining problem in dynamic graph. We introduced five new pattern types which rely on the extraction of dense subgraphs and the identification of their evolution. We formalized this task into a local-to-global framework: Local patterns are first mined in a static graphs; then they are combined with the one extracted in the previous graph to form evolving patterns. These patterns are defined by means of constraints that are used by the proposed algorithm Evolving-Subgraphs. It efficiently mines all evolving patterns that satisfy the constraints. Our experiments on real life datasets show that our approach produces high quality patterns that are useful to understand the graph dynamics.

References

[1] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over time: densification laws, shrinking diameters and possible explanations," in *Proc. KDD*. ACM, 2005, pp. 177–187.

[2] F. Zhu, X. Yan, J. Han, and P. S. Yu, "GPrune: A constraint pushing framework for graph pattern mining," in *Proc. PAKDD*, ser. LNCS, vol. 4426. Springer, 2007, pp. 388–400.

[3] W. Hämäläinen, H. Toivonen, and V. Poroshin, "Mining relaxed graph properties in internet," in *Proc. ICWI*. IADIS, 2004, pp. 152–159.

[4] J. Pei, D. Jiang, and A. Zhang, "On mining cross-graph quasi-cliques," in *Proc. KDD*. ACM, 2005, pp. 228–238.

[5] T. Uno, "An efficient algorithm for enumerating pseudo cliques," in *Proc. ISAAC*, ser. LNCS, vol. 4835. Springer, 2007, pp. 402–414.

[6] M. E. J. Newman, "Fast algorithm for detecting community structure in networks," *Phys. Rev. E*, vol. 69, no. 6, p. 66133, 2004.

[7] S. Fortunato and M. Barthelemy, "Resolution limit in community detection," *PNAS*, vol. 104, no. 1, pp. 36–41, 2007.

[8] U. Brandes, D. Delling, M. Gaertler, R. Görke, M. Hoefer, Z. Nikoloski, and D. Wagner, "On modularity clustering," *IEEE TKDE*, vol. 20, no. 2, pp. 172–188, 2008.

[9] L. De Raedt and A. Zimmermann, "Constraint-based pattern set mining," in *Proc. SDM*. SIAM, 2007, pp. 237–248.

[10] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol. 435, no. 7043, pp. 814–818, 2005.

[11] K. Makino and T. Uno, "New algorithms for enumerating all maximal cliques," in *Proc. SWAT*, ser. LNCS, vol. 3111. Springer, 2004, pp. 260–272.

[12] K. M. Borgwardt, H.-P. Kriegel, and P. Wackersreuther, "Pattern mining in frequent dynamic subgraphs," in *Proc. ICDM*. IEEE Computer Society, 2006, pp. 818–822.

[13] M. Lahiri and T. Y. Berger-Wolf, "Mining periodic behavior in dynamic social networks," in *Proc. ICDM*. IEEE Computer Society, 2008, pp. 373–382.

[14] F. Bonchi and C. Lucchese, "Extending the state-of-the-art of constraint-based pattern discovery," *Data Knowl. Eng.*, vol. 60, no. 2, pp. 377–399, 2007.

[15] F. Bodon, "A trie-based Apriori implementation for mining frequent item sequences," in *Proc. OSDM*. ACM, 2005, pp. 56–65.

[16] A. Chaintreau, J. Crowcroft, C. Diot, R. Gass, P. Hui, and J. Scott, "Pocket switched networks and the consequences of human mobility in conference environments," in *Proc. WDTN*. ACM, 2005, pp. 244–251.

[17] N. Eagle and A. Pentland, "Reality mining: Sensing complex social systems," *Journal of Personal and Ubiquitous Computing*, vol. 10, no. 4, pp. 255–268, 2006.