Cohesive Co-evolution Patterns in Dynamic Attributed Graphs

Elise Desmier[1], Marc Plantevit[2], Céline Robardet[1], and Jean-François Boulicaut[1]

[1] Université de Lyon, CNRS,
INSA-Lyon, LIRIS, UMR5205, F-69621, France
[2] Université de Lyon, CNRS,
Université Lyon 1, LIRIS, UMR5205, F-69622, France

Abstract. We focus on the discovery of interesting patterns in dynamic attributed graphs. To this end, we define the novel problem of mining cohesive co-evolution patterns. Briefly speaking, cohesive co-evolution patterns are tri-sets of vertices, timestamps, and signed attributes that describe the local co-evolutions of similar vertices at several timestamps according to set of signed attributes that express attributes trends. We design the first algorithm to mine the complete set of cohesive co-evolution patterns in a dynamic graph. Some experiments performed on both synthetic and real-world datasets demonstrate that our algorithm enables to discover relevant patterns in a feasible time.

1 Introduction

Real-world phenomena are often depicted by graphs where vertices represent entities and edges represent their relationships or interactions. With the rapid development of social media, sensor technologies and bioinformatics assay tools, large heterogeneous information networks have become available and deserve new knowledge discovery methods. As a result, graph mining has become an extremely active research domain. It has recently been extended into several complementary directions as multidimensional graphs [5], attributed graphs [16,17,22], and dynamic graphs [6]. Indeed, entities can be described by one or more attributes that constitute the attribute vectors associated with the graph vertices. Moreover, in many applications, edges may appear or disappear through time giving rise to dynamic graphs.

So far, sophisticated methods have been designed to provide new insights from attributed or dynamic graphs. Recent contributions have shown that using additional information associated to vertices enables to exploit both the graph structure and local vertex attributes [16,17,22]. Dynamic graphs have been studied in two different ways. On one hand, it is possible to study the evolution of specific properties (e.g., the diameter). On the other hand, it makes sense to look at local patterns and it provides a large spectra of approaches to characterize the evolution of graphs with association rules [3,18], transformation rules [24] or other types of patterns [6,12,13,15,20].

Analysing *dynamic attributed graphs* (i.e., sequence over time of attributed graphs whose relations between vertices and attributes values depends on the timestamp) has been less studied and we claim that it is interesting for several reasons. First, this kind of data offers a richer representation of real-world phenomena in which entities have their

J.-G. Ganascia, P. Lenca, and J.-M. Petit (Eds.): DS 2012, LNAI 7569, pp. 110–124, 2012.

own characteristics (vertex attributes). Furthermore, both entities and their interactions (edges) may evolve through time. Second, we believe that we cannot handle separately attributed graphs and dynamic graphs without introducing severe biases in the knowledge discovery process. Indeed, local vertex attributes and the role of the vertex within the graph are often closely related. Our thesis is that the simultaneous mining of the vertex attributes and the temporal dimensions has to be studied.

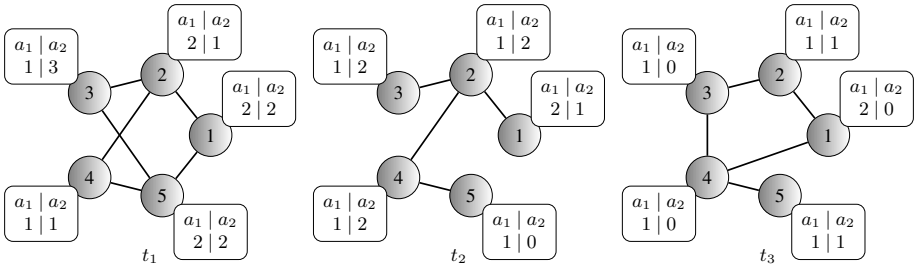


Fig. 1. Example of an undirected dynamic attributed graph

In this work, we assume that attributes related to vertices are numerical or ordinal. Let us illustrate our approach on a dynamic co-authorship graph depicted in Figure 1. This graph involves five vertices (i.e., authors labeled from 1 to 5) through three timestamps (t_1, t_2 and t_3). Each vertex/author is described with two numerical attributes (a_1 and a_2) that are indicators about each author. For instance, it could be the number of publications in a specific conference series during the period identified by the timestamp. It could also be the number of hours per week spent by the author on instructional duties. For each author, at each timestamp, we know the value of every attributes. As an example, the value of Attribute a_1 for Vertex 1 at Timestamp t_3 is 2. Co-evolution patterns are tri-sets of vertices, timestamps, and signed attributes that describe the local co-evolutions of similar vertices at several timestamps according to set of signed attributes that express attribute trends. In Figure 1, Vertices 1 and 3 share the same evolutions of their Attributes a_1 and a_2 over the three graphs: a_1 remains constant and a_2 decreases over time. Two vertices can be similar if they share a same neighborhood, i.e., they are close within the graph like Vertices 1 and 3 that share the same neighbors in the three timestamped graphs: Vertices 2 and 5 at Time t_1, Vertex 2 at Time t_2 and Vertices 2 and 4 at Time t_3. They can also be considered as similar if they play a similar role in the graph. This data mining problem is challenging since the search space is very large and clever enumeration strategies of all co-evolution patterns are needed when looking for complete extractions.

Our main contributions are as follows. We define the problem of mining co-evolution patterns in dynamic attributed graphs in Section 2 and we propose the first algorithm that computes them thanks to dedicated pruning strategies in Section 3. Then, in Section 4, we report an experimental study on two real datasets showing that this new kind of pattern holds in the data, that computing them in practice is feasible, and that they can be easily interpreted. The related work is discussed in Section 5 and Section 6 briefly concludes.

2 Problem Setting

2.1 A New Pattern Domain

Let us define a dynamic attributed graph $\mathcal{G}$ as a sequence over the time period $\mathcal{T}$ of attributed graphs $\langle G_1, \cdots, G_{|\mathcal{T}|} \rangle$ where each attributed graph G_t is a triplet $(V, E_t, \mathcal{A})$ with V as a set of vertices that is invariant with t, E_t as a set of not valued edges that depends on t, and $\mathcal{A}$ as a set of vertex attributes that gathers functions that associate a numerical value to the vertices of V at each timestamp $t \in \mathcal{T}$ [1]. More formally, $p \in \mathcal{A}$ is an application from $V \times \mathcal{T}$ to $\mathbb{R}$, i.e., $p(n, t)$ returns the value of attribute p for vertex n in timestamped graph G_t. To provide the formal definition of a co-evolution pattern, we need to define a relation between vertex attributes with regard to time. Let $s \in \{+, -, =\}$, we define $\rhd_s$ as follows:

- if s is equal to $+$, $\rhd_s$ stands for the relation $<$,
- if s is equal to $-$, $\rhd_s$ stands for the relation $>$,
- if s is equal to $=$, $\rhd_s$ stands for the relation $=$.

Definition 1 (co-evolution pattern). *Let us consider a dynamic attributed graph $\mathcal{G}$ and a triplet (N, T, P) such that $N \subseteq V$ is a set of vertices, $T \subset \mathcal{T}$ is a set of not necessarily consecutive timestamps and P is a set of signed attributes, that is a subset of $\mathcal{A} \times \rhd$. (N, T, P) is a co-evolution pattern of $\mathcal{G}$ iff: (i) $\forall n \in N$, $\forall t \in T$ and $\forall (p, s) \in P$, $p(n, t) \rhd_s p(n, t + 1)$. (ii) (N, T, P) is maximal: adding any vertex, any timestamp or any signed attribute leads to the violation of (i).*
We denote by $evol(N, T, P)$ the fact that (N, T, P) is a co-evolution pattern.

For the sake of simplicity, we also denote the pair (p, s) by p^s. Considering the data from Figure 1, three examples of co-evolution patterns are $\big(\{1, 3\}, \{t_1, t_2\}, \{a_1^=, a_2^-\}\big)$, $(\{1, 3, 4\}, \{t_1, t_2\}, \{a_1^=\})$ and $\big(\{1, 2, 3, 4\}, \{t_2\}, \{a_1^=, a_2^-\}\big)$. Let us notice that the presence of a timestamp t in T means that condition $\rhd_s$ on every p^s for every vertex holds between t and t+1. Thus, the last timestamp of $\mathcal{G}$ cannot appear in a co-evolution pattern.

In many real-world applications, it is difficult to obtain strict equality of an attribute value between two consecutive timestamps, for instance when the values come from sensors. Thus, it is important to relax such a condition. To this end, we consider similarity between attribute values. Given a threshold $\delta \in [0, 1]$, $p(n, t)$ and $p(n, t + 1)$ being two values of Attribute p of Vertex n at Timestamp t and $t + 1$, are said *similar* (denoted as $p(n, t) \rhd_=^\delta p(n, t + 1)$) if $(1 - \delta)p(n, t) < p(n, t + 1) < (1 + \delta)p(n, t)$. We can easily derive the definitions for $\rhd_+^\delta$ and $\rhd_-^\delta$:

$$(1 + \delta)p(n, t) < p(n, t + 1) \Leftrightarrow p(n, t) \rhd_+^\delta p(n, +1),$$
$$(1 - \delta)p(n, t) > p(n, t + 1) \Leftrightarrow p(n, t) \rhd_-^\delta p(n, t + 1).$$

We are interested in specific co-evolution patterns that satisfy some user-defined relevancy constraints: we look for patterns that are somehow "large" enough and that gather "cohesive" vertices. Let us now formalize such notions.

[1] The collection $\mathcal{A}$ does not change while the values of these attributes depend on the relative vertices and timestamps.

2.2 Volume

A co-evolution pattern (N, T, P) is said to be *large* when its volume is greater than or equal to a given threshold ϑ. We propose the following measure to compute the so-called natural volume of a pattern: $volume(N, T, P) = (|N|)^{\gamma_v} \times (|T|)^{\gamma_t} \times (|P|)^{\gamma_p}$ This definition of the volume is a simple product of set cardinalities (i.e., $\gamma_v = \gamma_t = \gamma_p = 1$). However, in practice the sizes of V, $\mathcal{T}$ and $\mathcal{A}$ are of different orders of magnitude. Indeed, a dynamic attributed graph often contains much more vertices than timestamps or attributes. This may lead to favor patterns which have a larger number of elements within the largest dimension (often V). Therefore, it makes sense to correct this measure to focus on patterns that have more elements from smaller dimensions, i.e., adding an element to the largest dimension should lead to a corrected volume smaller than adding an element to the other dimensions. To this end, the exponents are defined as follows: $\gamma_v = \frac{|V|+(|\mathcal{T}|-1)+|\mathcal{A}|}{3\times|V|}$, $\gamma_t = \frac{|V|+(|\mathcal{T}|-1)+|\mathcal{A}|}{3\times(|\mathcal{T}|-1)}$ and $\gamma_p = \frac{|V|+(|\mathcal{T}|-1)+|\mathcal{A}|}{3\times|\mathcal{A}|}$.

Considering our running example: $volume(\{1,3,4\}, \{t_1, t_2\}, \{a_1^=\}) = 3^{\frac{9}{15}} \times 2^{\frac{3}{2}} \times 1^{\frac{3}{2}} = 5.47$, while the natural volume (i.e., $\gamma_i = 1$) is $3 \times 2 \times 1 = 6$.

2.3 Taking into Account the Graph Structure

Until now, the specification of a co-evolution pattern only considers a set of vertices that follow the same evolution among a set of signed attributes over a set of timestamps. It is then important to take into account the graph structure, i.e., the edges of the graph at every timestamp. To this end, we use similarity measure between vertices.

Definition 2 (Cohesive co-evolution pattern). *Given a similarity threshold $\sigma \in [0, 1]$ and a similarity measure sim, a co-evolution pattern (N, T, P) is said to be cohesive if the following condition holds:*

$$cohesive(N, T, P) \equiv \forall t \in T,\ \forall u, v \in N^2,\ sim(u, v, G_t) \geq \sigma$$

Any similarity measure can be considered. Let us introduce some of them that exploit differently the graph structure. We can focus on the direct neighborhood of vertices to assess how they are similar. To this end, cosine or Jaccard measures are well adapted. Let $Adj(u)$ denotes the set of vertices that are adjacent to Vertex u.

$$cos(u, v) = \frac{|\,(Adj(u) \cup \{u\}) \cap (Adj(v) \cup \{v\})\,|}{\sqrt{|Adj(u) \cup \{u\}| \times |Adj(v) \cup \{v\}|}}$$

$$jac(u, v) = \frac{|\,(Adj(u) \cup \{u\}) \cap (Adj(v) \cup \{v\})\,|}{|Adj(u) \cup \{u\} \cup Adj(v) \cup \{v\}|}$$

These definitions require to take into account vertices that are adjacent to Vertices u and v but also Vertices u and v themselves to make it possible to have perfect similarity (i.e., similarity equals to 1) even if self loops are not allowed. Cosine and Jaccard measures only consider the direct neighborhood, i.e., the adjacent vertices. However, one may exploit larger neighborhood to establish similarity based on spreading activation. It enables to highlight vertices that play similar roles within the graph. Thiel and Berthold have introduced this kind of measure in [23]. They introduced two types of similarity

both based on spreading activation. Thus the similarity between two vertices needs to consider and compare the graphs after diffusion of the activation with each of the two vertices as spreading starts.

$$a_v^0(u) = 0, \text{ if } v \neq u, \ 1 \text{ otherwise}$$
$$a_v^k(u) = \frac{\sum_{i \in Adj(v)} a_i^{k-1}(u)}{|| \sum_{i \in Adj(v)} a_i^{k-1}(u)||_2}$$

The activation of the graph is associated to a beginning activation Vertex u and is computed in k_{max} steps. $a^k(u)$ represents the vector of activation of the graph at Step k of the spreading started at u and $a_v^k(u)$ represent the activation of Vertex v, i.e., the value of the Vertex v in the vector with $0 < k < k_{max}$. At Step 0, the only activated vertex is u with a value of 1. Thus, at the following Steps $k \in [1, k_{max}]$, each vertex activation depends on its direct neighbor activation value at $k - 1$.
Two similarity measures are now defined. The first one, called *the activation similarity*, compares the result of this spreading on the k-neighborhood, and the common activated vertices.

$$\hat{a}^\star(u) = D^{-1/2} \sum_{k=0}^{k_{max}} \alpha^k a^k(u)$$
$$Activation = cos(\hat{a}^\star(u), \hat{a}^\star(v)) = \frac{\sum_{i=1}^{n} \hat{a}_i^\star(u) \hat{a}_i^\star(v)}{||\hat{a}^\star(u)|| \, ||\hat{a}^\star(v)||}$$

with α a decay parameter of a^k in the final result and D the degree matrix. The degree matrix is a diagonal matrix which contains the degree of each vertex in the graph.

As the similarity here is looking for common close neighborhood, the spreading activation needs to loose weight over spreading with a certain value α. The more u and v have common highly activated vertices, the higher is the activation similarity $cos(\hat{a}^\star(u), \hat{a}^\star(v))$. The second measure, called *the signature similarity*, compares how activation spreads into the neighborhood and then compares the structure of the vertices neighborhood.

$$\delta^k(u) = \begin{cases} 0, \text{ if } k = 0 \\ a^k(u) - a^{k-1}(u) \end{cases}$$
$$\tau^k(u) = ||\delta^k(u)||_2$$
$$Signature = cos(\tau(u), \tau(v)) = \frac{\sum_{k=1}^{kmax} ||\delta^k(u)| \, ||\delta^k(v)||}{||\tau(u)|| \, ||\tau(v)||}$$

The velocity vector $\delta^k(u)$ represents the change of activation of the graph between spreading steps and the convergence vector $\tau^k(u)$ describes the convergence speed in the spreading process. The more the structure of the vertices neighborhood is similar, the higher is the signature similarity $cos(\tau(u), \tau(v))$. Thanks to these two similarities, we can compare a larger neighborhood than with the cosine or Jaccard measures. Moreover, the signature similarity can be considered as a role comparison and it allows completely different patterns.

2.4 Mining Task

The problem we aim to solve is defined as follows:

Problem 1 (Cohesive co-evolution pattern mining). Let $\mathcal{G}$ be a dynamic attributed graph. Given a similarity measure sim, a minimum vertex similarity threshold σ and a minimum volume threshold ϑ, the goal of mining cohesive co-evolution patterns is to find the complete set of co-evolution patterns that have a volume greater or equal to ϑ and fulfill the cohesiveness constraint.

In the following section, we define an algorithm that efficiently mines all large cohesive co-evolution patterns and only them.

3 Mining Cohesive Co-evolution Patterns

The enumeration of all the patterns by materializing and traversing all possible tri-sets from $V \times \mathcal{T} \times (\mathcal{A} \times \triangleright)$ is in practice not feasible. Therefore, we look for a decomposition of the original search space into smaller pieces such that each portion can be independently studied in main memory and such that the union of the cohesive co-evolution patterns extracted from each portion is the whole collection of cohesive co-evolution patterns. Therefore, all possible tri-sets are explored in a depth-first search manner. We use a binary enumeration inspired from the strategy for closed set mining in tensors described in [7].

At each step, a tri-set (N, T, P) from $V \times \mathcal{T} \times (\mathcal{A} \times \triangleright)$ is considered and can be represented as a node in the binary enumeration tree. Each node contains the elements that have already been enumerated and added or not to the current pattern. Given elements already enumerated, the node also maintains candidate elements, i.e., elements that can be potentially added to the current pattern. To this end, we define, for each dimension, three sets to summarize the enumeration choices made before the current tri-set (N, T, P) and to maintain the candidate elements to be enumerated:

- $In.N$, $In.T$ and $In.P$ respectively denote the vertices, the timestamps and the signed attributes that have already been enumerated and chosen as being in (N, T, P), i.e., $In.N = N$, $In.T = T$, and $In.P = P$.
- $Out.N$, $Out.T$ and $Out.P$ respectively denote the vertices, the timestamps and the signed attributes that have already been enumerated but were not added to (N, T, P).
- $Cand.N$, $Cand.T$ and $Cand.P$ respectively denote the candidates vertices, timestamps and signed attributes, i.e., elements that are promising for enlarging the current pattern (N, T, P).

Therefore, at a node n an element e from $Cand = Cand.N \cup Cand.T \cup Cand.P$ is selected and two nodes are generated, one by adding e to $In = In.N \cup In.T \cup In.P$, the other one by adding e to $Out = Out.N \cup Out.T \cup Out.P$. The process is recursively iterated until $Cand$ becomes empty. This enumeration is correct and complete:

it computes the complete set of cohesive co-evolution patterns. Its efficiency relies on safe pruning possibilities thanks to constraints:

Volume: This constraint can be easily exploited. Indeed, given the pattern (N, T, P), if the volume constraint does not hold and adding all candidates from $Cand$ does not enlarge enough the pattern to satisfy it, the enumeration can be safely interrupted.

Cohesiveness: Given (N, T, P), we can propagate this constraint among $Cand.N$, $Cand.T$, $Out.N$, and $Out.T$. We can discard vertices from $Cand.N$ and $Out.N$ that are too different than vertices from N at a given time from T, and we can remove timestamps from $Cand.T$ and $Out.T$ when vertices from N are too different.

Evolution of properties: This constraint can be propagated to discard $Cand$ and Out vertices, timestamps and properties that do not follow the evolution of (N, T, P). First, in $Cand.P$ and $Out.P$ we can remove signed attributes that are in contradiction with those in P. For instance, if $X^+ \in In.P$, we delete X^- and $X^=$ from $Cand.P$ and $Out.P$. We can also remove signed attributes whose trends are not followed by vertices from N at a given timestamp from T. Second, we can remove vertices from $Cand.N$ and $Out.N$ that do not follow trends of P at a given timestamp from T. Finally we can remove timestamps from $Cand.T$ and $Out.T$ when vertices from N do not follow trends of P.

Maximality: We can exploit this constraint by checking if In can still be maximal or if the existence of an enumerated element not added within the pattern prevent it to be maximal. Let $\cup_N^{IC} \equiv In.N \cup Cand.N$, $\cup_T^{IC} \equiv In.T \cup Cand.T$, $\cup_P^{IC} \equiv In.P \cup Cand.P$, a pattern (N, T, P) *can be maximal* ($canBeMaximal\,(In, Cand, Out)$) if the following set of conditions are satisfied:

- $\forall v \in Out.N$, $\neg cohesive\left(v, \cup_N^{IC}, \cup_T^{IC}\right) \vee \neg evol\left(v, \cup_T^{IC}, \cup_P^{IC}\right)$
- $\forall t \in Out.T$, $\neg cohesive\left(\cup_N^{IC}, \cup_N^{IC}, t\right) \vee \neg evol\left(\cup_N^{IC}, t, \cup_P^{IC}\right)$
- $\forall p \in Out.P$, $\neg evol\left(\cup_N^{IC}, \cup_T^{IC}, p\right)$

Algorithm 1 presents the general enumeration principle. Notice that the pruning operations are different with respect to the type of the element to be enumerated (see Lines 12, 16 and 19, cohesiveness and evolution pruning). For the sake of simplicity, e will denote either a vertex, a timestamp or an attribute. Details of these subroutines are:

PruneAddingVertex: The addition of a vertex e to $In.N$ makes inconsistent some elements of $Cand.N, Cand.T, Cand.P$ that have to be removed. We also filter elements from $Out.N, Out.T$ and $Out.P$ to further verify the predicate $canBeMaximal$:

$Cand.N \leftarrow \{v \in Cand.N \setminus \{e\} \,|cohesive(e, v, In.T)\}$
$Out.N \leftarrow \{v \in Out.N|cohesive(e, v, In.T)\}$
$Cand.T \leftarrow \{t \in Cand.T|cohesive(e, In.N, t) \wedge evol(e, t, In.P)\}$
$Out.T \leftarrow \{t \in Out.T|cohesive(e, In.N, t) \wedge evol(e, t, In.P)\}$
$Cand.P \leftarrow \{p \in Cand.P|evol(e, In.T, p)\}$
$Out.P \leftarrow \{p \in Out.P|evol(e, In.T, p)\}$

PruneAddingTime: The addition of a timestamp enables to remove elements from $Cand.N$ and $Cand.P$. We also filter elements from $Out.N$ and $Out.P$:

Algorithm 1. PatternExtraction

Input: $Cand$,In,Out,σ, ϑ
Output: A set of cohesive co-evolution patterns

```
1  begin
2      inIsOk ← false;
3      if (volume(In.N ∪ Cand.N,In.T ∪ Cand.T,In.P ∪ Cand.P)≥ ϑ)∧
       CanBeMaximal(Cand,In,Out) then
4          if isEmpty(Cand) then
5              output In.N,In.T,In.P;
6          else
7              e ← chooseElement(Cand);
8              Cand ← Cand \ {e};
9              PatternExtraction(Cand,In,Out ∪ {e},ϑ);
10             In ← In ∪ {e};
11             if e ∈ N then
12                 (Cand, Out) ← PruneAddingVertex(e, In, Cand, Out);
13                 inIsOk ← cohesive(e, In.N, In.T) ∧ evol(e, In.T, In.P);
14             else
15                 if e ∈ T then
16                     (Cand, Out) ← PruneAddingTime(e, In, Cand, Out);
17                     inIsOk ← cohesive(In.N, In.N, e) ∧ evol(In.N, e, In.P);
18                 else if e ∈ P then
19                     (Cand, Out) ← PruneAddingProperty(e, In, Cand, Out);
20                     inIsOk ← evol(In.N, In.T, e);
21             if inIsOk then
22                 PatternExtraction(Cand,In,Out,σ,ϑ);
```

$Cand.N \leftarrow \{v \in Cand.N | cohesive(In.N, v, e) \wedge evol(v, e, In.P)\}$
$Out.N \leftarrow \{v \in Out.N | cohesive(In.N, v, e) \wedge evol(v, e, In.P)\}$
$Cand.P \leftarrow \{p \in Cand.P | evol(In.N, e, p)\}$
$Out.P \leftarrow \{p \in Out.P | evol(In.N, e, p)\}$

PruneAddProperty: The addition of a signed attribute leads to the following filtering:
$Cand.P \leftarrow Cand.P \setminus \{e\}$
$Out.P \leftarrow Out.P \setminus \{e\}$
$Cand.N \leftarrow \{v \in Cand.N | \neg evol(v, In.T, e)\}$
$Out.N \leftarrow \{v \in Out.N | \neg evol(v, In.T, e)\}$
$Cand.T \leftarrow \{t \in Cand.T | \neg evol(In.N, t, e)\}$
$Out.T \leftarrow \{t \in Out.T | \neg evol(In.N, t, e)\}$

Table 1. Main characteristics of the dynamic attributed graphs

DATASET	$\#V$	$\#\mathcal{T}$	$\#\mathcal{A}$	Avg. density
synthetic	500	10	20	0.164
DBLP	2,723	9	43	0.002
Brazil landslide	10,521	2	8	0.0065

4 Experimental Study

Let us now report on experimental results for both synthetic and real-world datasets. We first describe the used dynamic attributed graphs, then we provide quantitative and qualitative results. All experiments were performed on a cluster. Nodes are equipped with 16 processors at 2.5GHz and 16GB of RAM under Linux operating systems. The algorithm has been implemented in C++ and compiled with GCC 4.1.2.

4.1 Dataset Description

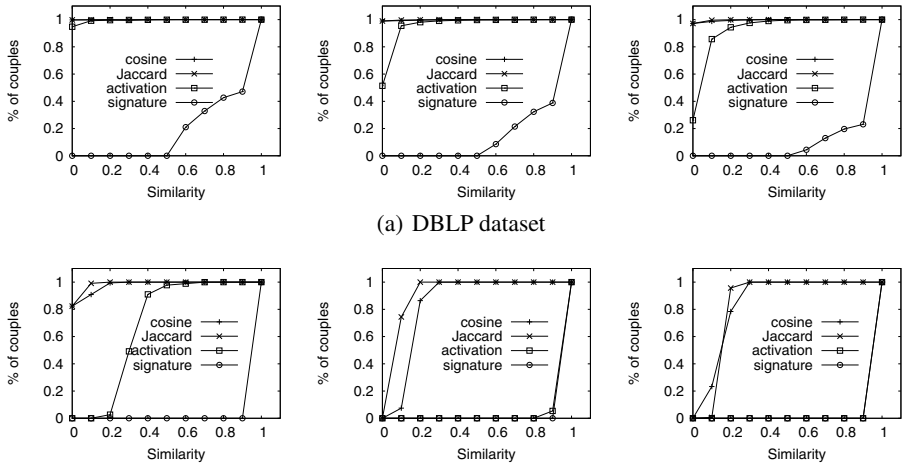
We consider one synthetic and two real-world dynamic attributed graphs whose characteristics are given in Table 1.

Synthetic: We generated an Erdös-Rényi graph with n vertices and a uniform probability p_0 that there is an edge between two vertices of 0.04 for time t_0. Notice that this density will change for each timestamp, consequently, the average density presented in Table 1 for the whole dataset is not 0.04. For time $t \in [t_1, t_{|\tau|-1}]$, we built the graphs G_t from the graph G_{t-1} by introducing some edge-perturbations. To this end, we introduced the probability of edge-change, that is the probability that an edge which does not belong to the set of edges at time $t-1$ appears at time t (and vice-versa for edge-disappearance). This probability was set to 0.04. For attributes related to vertices, we generated a random values between 0 and 100 at Time t_0 for each vertex. Then, for each Time t_i, with $i \in [1, |\tau| - 1]$, we introduced attributes variation probabilities, i.e., the probability of increase p_+ and the probability of decrease p_-.

DBLP: We consider a subset of the DBLP[2] dataset. Vertices of the graph are authors and an edge exists between them if the corresponding authors have written a paper together in a given period of time. Only authors who had at least 10 publications (in a selected set of 43 conferences/journals) from 1990 to 2010 are considered. There are in total 2,723 authors. Each graph depicts co-authorship relations over 5 years ([1990-1994][1992-1996]...[2006-2010]). Notice that we consider overlapping time periods to maintain a coherence in the authorship relations. Each vertex at each time is associated to a set of 43 attributes corresponding to the number of publications in each conference/journal during the related period. To summarize, this dataset consists in 2,723 vertices, 9 timestamps and 43 attributes. This dataset is singular as there is a large scale of conferences/journals in which each author will have a significative set with no publication all along their carrier. Thus in the experiments, equal evolution (i.e., $\triangleright_=$) will not be considered.

Brazil landslide: This dataset is extracted from two satellite images taken before and after a huge landslide. 10,521 regions (i.e., shapes in the image) have been computed with 9 attributes from a picture of 250,000 square meters of ground. The segmentation was performed in eCognition 8.64 with a scale factor of 20 [2]. Therefore, a vertex is a region (segmented area) and there is an edge between two vertices if they are at less than 50 pixels (25m). Having only two timestamps, we aim at looking for significant attributes variations that could characterize a landslide.

[2] DBLP is a computer science bibliography: `http://dblp.uni-trier.de/`



(a) DBLP dataset

(b) synthetic dataset

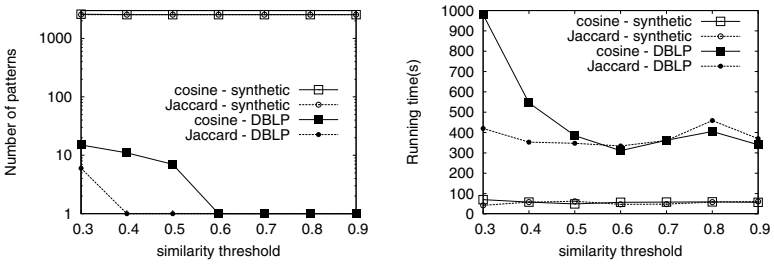
Fig. 2. Cumulative distributions of vertex similarities at first, middle and last time

We report in Figure 2 the cumulative distribution of similarities between vertices for synthetic and DBLP datasets. Notice that each measure has his own distribution. Activation and signature similarities need two parameters. In [23], the authors used $\alpha = 0.3$ and they recommend not to have k higher than 10: we choose $\alpha = 0.3$ and $k = 5$.

4.2 Quantitative Results

Figure 3(a) and Figure 3(b) present, respectively, the number of patterns and relative execution times with regard to the similarity threshold, for cosine and Jaccard similarities in DBLP and synthetic datasets. Considering the synthetic dataset, notice that the number of patterns remains stable. This is mainly due to the fact that most of the similarities of vertices couples is lower than the considered threshold. Consequently, most of the discovered patterns contain only one vertex. This explains that the related execution times for this dataset remain constant. In DBLP dataset the impact of this threshold is more important since the running time increases when the similarity constraint becomes less stringent. Figure 4(a), 4(b) and 4(c) also report relative execution times with regards to the similarity threshold, for activation and signature similarities in DBLP and synthetic datasets. Notice that the similarity threshold meets "harder distribution", i.e., vertices couples similarities are stronger than Jaccard and cosine ones. The execution times are highly related to the distribution of vertex couples similarities and thus to the similarity threshold.

Figure 4(d) and 4(e) show the execution times of our algorithm according to the volume threshold for activation and Jaccard Similarities in the synthetic dataset. The running time decreases when the volume threshold increases, meaning that our algorithm enables to prune large parts of the search space.



(a) (b)

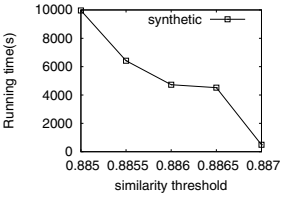
Fig. 3. Comparing similarity thresholds for cosine and Jaccard for 2 datasets ($\vartheta = 20$)

4.3 Qualitative Results

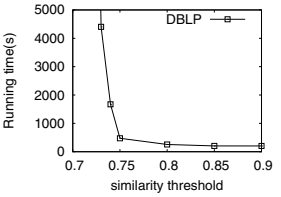
We examine the patterns obtained on the DBLP dataset without the equality evolution. Considering the Jaccard similarity, Table 2 reports two patterns having the most important number of timestamps whose vertices have at least 10% of common neighborhood. The first cohesive co-evolution pattern depicts a set of nine authors close in the co-autorship graph. Their respective number of publications in VLDB conference series decrease whereas their number of publications in PVLDB increases between 2002-2006 and 2004-2008 and between 2004-2008 and 2006-2010. For the sake of simplicity, let us say the number of publications increases/decreases between 2002 and 2010. This pattern reflects the new policy of the VLDB endowment. Indeed, PVLDB appeared in 2008, the review process of the VLDB conference series is done in collaboration with in 2010 and entirely through PVLDB in 2011. The second pattern describes a group of 4 authors who have an increasing number of publications in top data mining/databases conferences between 1998 and 2006. These authors are major actors in the data mining community. Notice that the patterns extracted with the activation measure are similar to those extracted with Jaccard ones.

Table 3 presents a pattern extracted using the signature similarity with a similarity threshold of 0.999 and a volume threshold of 30. This pattern identifies rising authors in the bioinformatics area (understood as publishing in journals *Bioinformatics* and *BMC Bioinformatics*). Indeed, they have none or few publications in the two journals during the period 2000-2004 and more publications within the period 2002-2006. Notice that these authors do not have the same co-authors thus this pattern can not be discovered considering a Jaccard, cosine or activation measures.

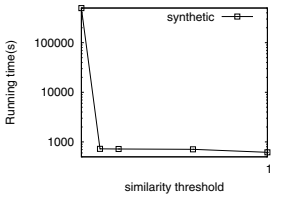
For the Brazil landslide dataset, we aim at discovering patterns that characterize landslides. We ran experiments with the Jaccard similarity ($\sigma = 0.5$) to discover sets of vertices that are in a similar neighborhood and have some attributes that significantly change between the two timestamps ($\delta = 0.7$). Preliminary experiments seem to be promising. If we select patterns which contain the attribute NDVI (a vegetation index) we obtain those represented in white in Figure 5. The Groups of regions 1 and 4 are actually landslides while the Groups 2, 3 and 5 do not. When looking at the original data, we can see a shift between the two pictures, because of which the Groups of regions 2, 3 and 5 contain vegetation boarding the road in the first picture and a road in the second one.



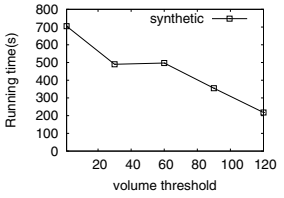
(a) synthetic activation similarity ($\vartheta = 30$)



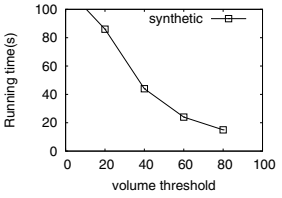
(b) DBLP activation similarity ($\vartheta = 30$)



(c) synthetic signature similarity ($\vartheta = 30$)



(d) synthetic activation similarity ($\sigma = 0.75$)



(e) synthetic Jaccard similarity ($\sigma = 0.21$)

Fig. 4. Running times of the different experiments

Table 2. Patterns extracted from DBLP with the Jaccard similarity ($\sigma = 0.1$, $\vartheta = 30$)

N	T	P	
Authors	Time steps	Decreasing publications	Increasing publications
Jeffrey F.Naughton, Hector Garcia-Molina, Joseph M. Hellerstein, Gerhard Weikum, David J. DeWitt, Stanley B. Zdonik, Michael Stonebraker, Serge Abiteboul, Michael J. Franklin	2002-2006 2004-2008	VLDB	PVLDB
Jianyong Wang, Ke Wang, Jiawei Han, Jian Pei	1998-2002 2000-2004	-	TKDE, ICDE, KDD, SDM

Table 3. Pattern extracted from DBLP with the signature similarity ($\sigma = 0.999$, $\vartheta = 30$)

N	T	P
Authors	Time steps	Increasing publications
Debashis Gosh, Thomas Mailund, Jotun Hein, Gordon K. Smyth, Shuangge Ma, Jan A. Kors, Michael Q.Zhang, Sandor Pongor, Olivier Poch, Jong Bhak, Yudi Pawitan, Steven J.M. Jones, Jonas S. Almeida, Wei Pan, Wen-Lian Hsu, Hiroyuki Toh, Jianping Hua, Alessandro Sette, Falk Schreiber	2000-2004	Bioinformatics, BMC Bioinformatics

5 Related Work

Graph mining is well studied in data mining. In the literature, there exists two ways to analyse graphs. On one hand, graphs are studied by means of clustering techniques [11,9,4,10]. On the other hand, pattern mining allows the extraction of interesting patterns describing some interesting behavior. This has been applied both on dynamic and attributed graphs.

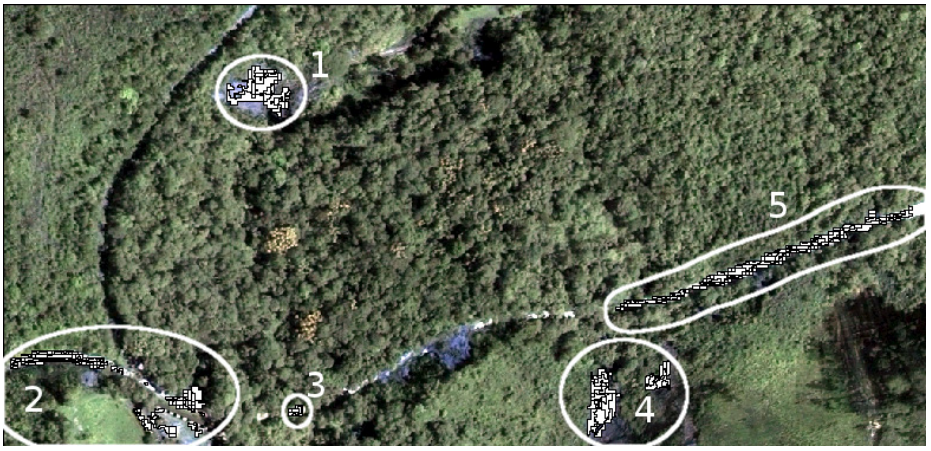


Fig. 5. Patterns extracted from Brazil landslide dataset with Jaccard similarity ($\sigma = 0.5$, $\delta = 0.7$)

Pattern mining in dynamic graphs is extensively studied. In [6], Borgwardt et al. define frequent dynamic subgraph mining (i.e., looking for subgraphs appearing similarly in consecutive times). Following the same idea, [15] mines subgraph appearing at periodic timestamps. Inokuchi and Washio [12] extract frequent induced subgraph subsequences such that a graph is considered as subgraph of another if there is an injective function on vertices, edges, labels and graphs of the sequence. In [19], Prado et al. propose an algorithm dedicated to mine frequent dynamic plane subgraphs from a database of plane graphs. These patterns then can be used as the basis for the extraction of what the authors called spatiotemporal patterns. All these methods are looking for patterns that are somehow stable or preserved but they do not consider patterns on evolutions. On the contrary [1] mines the evolution of conserved relational states (i.e., sequences of consecutive time-conserved patterns sharing a minimum number of vertices). Also, [20] proposes the extraction of evolving patterns such that pseudo-cliques of consecutive timestamps are related if they have a temporal event relationship. An other way to characterize a graph is the extraction of rules. Berlingerio et al. [3] introduce graph evolution rules based on frequency time patterns and in [18], the authors propose multidimensional association rules. [24] studies how a graph is structurally transformed through time. The proposed method computes graph rewriting rules that describe the evolution between consecutive graphs. These rules are then abstracted into patterns representing the dynamics of graphs.

In parallel, static attributed graph have been widely studied as well. Moser et al. [16] pioneered this topic proposing a method to find dense homogeneous subgraphs (i.e., whose vertices share a large set of attributes). Silva et al. [22] extract pairs of dense subgraphs and boolean attribute sets such that the attributes are strongly associated with the subgraphs. The authors of [17] introduce the task of finding collections of homogeneous k-clique percolated components (i.e., made of overlapping cliques sharing a common set of attributes). A larger neighborhood is considered in [14] where the authors relax the constraints on the structure while extracting proximity patterns. Roughly speaking, they propose a probabilistic approach to both construct the neighborhood of a vertex

and to propagate information into this neighborhood. Following the same motivation, Sese et al. [21] extract (not necessarily dense) subgraphs with common itemsets. Note that all these methods only use one topological information based on the neighborhood, although it may be larger or less strict depending on the methods. Moreover, they do not handle numerical attributes.

In [13], Jin et al. introduce the *Trend Motif* approach. Their objectives are quite similar to ours: they consider weighted vertices, they aim at analyzing the dynamics of a network by discovering connected subgraphs whose vertex weights follow the same evolution. The evolution of weight is limited to increase and decrease on consecutive timestamps. Our approach is much more general since vertex attributes are not reduced to a singleton and timestamps in co-evolution patterns are not necessarily consecutive. Furthermore, notice that given a co-evolution pattern (N, T, P), for each $t \in T$, the subgraph induced on the complete graph G_t by N may be not connected.

6 Conclusion

Designing new methods to discover patterns gathering the dynamics of a graph is a timely challenge. Recently, such methods were proposed for the extraction of different kinds of patterns or rules in dynamic graphs (see for instance [3,8,12,18,20]). This work investigates a new direction in dynamic graph mining. We take into account the fact that attributes are often related to vertices in dynamic graphs. First, we have defined the novel problem of mining cohesive co-evolution patterns in the so-called dynamic attributed graphs. Then, we have designed and implemented a complete algorithm that computes them in a feasible time. We have reported an experimental study on both synthetic and real-world datasets. Building a global model by means of these patterns to summarize dynamic attributed graphs is an interesting topic that we may consider in the near future.

Acknowledgement. The authors thank ANR for supporting this work through the FOSTER project (ANR-2010-COSI-012-02). They also acknowledge support from the CNRS/IN2P3 Computing Center and the LSIIT laboratory for providing and preprocessing the data from Brazil.

References

1. Ahmed, R., Karypis, G.: Algorithms for Mining the Evolution of Conserved Relational States in Dynamic Networks. In: ICDM, pp. 1–10. IEEE (2011)
2. Baatz, M.: Multiresolution segmentation: an optimization approach for high quality multi-scale image segmentation. In: AGIT, vol. 58, pp. 12–23. Herbert Wichmann Verlag (2000)
3. Berlingerio, M., Bonchi, F., Bringmann, B., Gionis, A.: Mining Graph Evolution Rules. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009, Part I. LNCS, vol. 5781, pp. 115–130. Springer, Heidelberg (2009)
4. Berlingerio, M., Coscia, M., Giannotti, F., Monreale, A., Pedreschi, D.: As Time Goes by: Discovering Eras in Evolving Social Networks. In: Zaki, M.J., Yu, J.X., Ravindran, B., Pudi, V. (eds.) PAKDD 2010, Part I. LNCS, vol. 6118, pp. 81–90. Springer, Heidelberg (2010)
5. Berlingerio, M., Coscia, M., Giannotti, F., Monreale, A., Pedreschi, D.: Foundations of multidimensional network analysis. In: ASONAM, pp. 485–489. IEEE (2011)

6. Borgwardt, K.M., Kriegel, H.-P., Wackersreuther, P.: Pattern mining in frequent dynamic subgraphs. In: ICDM, pp. 818–822. IEEE (2006)
7. Cerf, L., Besson, J., Robardet, C., Boulicaut, J.F.: Closed patterns meet n-ary relations. TKDD 3(1), 3:1–3:36 (March 2009)
8. Cerf, L., Nguyen, T.B.N., Boulicaut, J.-F.: Discovering Relevant Cross-Graph Cliques in Dynamic Networks. In: Rauch, J., Raś, Z.W., Berka, P., Elomaa, T. (eds.) ISMIS 2009. LNCS, vol. 5722, pp. 513–522. Springer, Heidelberg (2009)
9. Cheng, H., Zhou, Y., Yu, J.X.: Clustering large attributed graphs: A balance between structural and attribute similarities. TKDD 5(2), 12:1–12:33 (2011)
10. Ester, M., Ge, R., Gao, B.J., Hu, Z., Ben-moshe, B.: Joint Cluster Analysis of Attribute Data and Relationship Data: the Connected k-Center Problem, pp. 246–257. SIAM (2006)
11. Günnemann, S., Kremer, H., Laufkötter, C., Seidl, T.: Tracing Evolving Subspace Clusters in Temporal Climate Data. In: DMKD, vol. 24, pp. 387–410. Springer (September 2012)
12. Inokuchi, A., Washio, T.: Mining frequent graph sequence patterns induced by vertices. In: SDM, pp. 466–477. SIAM (2010)
13. Jin, R., Mccallen, S., Almaas, E.: Trend Motif: A Graph Mining Approach for Analysis of Dynamic Complex Networks. In: ICDM, pp. 541–546. IEEE (2007)
14. Khan, A., Yan, X., Wu, K.L.: Towards proximity pattern mining in large graphs. In: SIGMOD, pp. 867–878. ACM (2010)
15. Lahiri, M., Berger-Wolf, T.Y.: Mining periodic behavior in dynamic social networks. In: ICDM, pp. 373–382. IEEE (2008)
16. Moser, F., Colak, R., Rafiey, A., Ester, M.: Mining cohesive patterns from graphs with feature vectors. In: SDM, pp. 593–604. SIAM (2009)
17. Mougel, P.-N., Rigotti, C., Gandrillon, O.: Finding Collections of *k*-Clique Percolated Components in Attributed Graphs. In: Tan, P.-N., Chawla, S., Ho, C.K., Bailey, J. (eds.) PAKDD 2012, Part II. LNCS, vol. 7302, pp. 181–192. Springer, Heidelberg (2012)
18. Nguyen, T.K.N., Cerf, L., Plantevit, M., Boulicaut, J.-F.: Multidimensional Association Rules in Boolean Tensors. In: SDM, pp. 570–581. SIAM (2011)
19. Prado, A., Jeudy, B., Fromont, E., Diot, F.: Mining spatiotemporal patterns in dynamic plane graphs. IDA Journal 17(1) (to appear, 2013)
20. Robardet, C.: Constraint-Based Pattern Mining in Dynamic Graphs. In: ICDM, pp. 950–955. IEEE (2009)
21. Sese, J., Seki, M., Fukuzaki, M.: Mining networks with shared items. In: CIKM, pp. 1681–1684. ACM (2010)
22. Silva, A., Meira Jr., W., Zaki, M.J.: Mining attribute-structure correlated patterns in large attributed graphs. PVLDB 5(5), 466–477 (2012)
23. Thiel, K., Berthold, M.R.: Node Similarities from Spreading Activation. In: ICDM, pp. 1085–1090. IEEE (2010)
24. You, C.H., Holder, L.B., Cook, D.J.: Learning Patterns in the Dynamics of Biological Networks. In: KDD, pp. 977–985. ACM (2009)