HABILITATION A DIRIGER DES RECHERCHES

présentée devant

l'Institut National des Sciences Appliquées de Lyon
et l'Université Claude Bernard LYON I

Constraint-based pattern mining approaches for the analysis of relational attributed dynamic graphs

SPECIALITE : INFORMATIQUE

Céline ROBARDET - 9 décembre 2013

Jean-François Boulicaut, Professeur, INSA de Lyon	Examinateur
Fosca Giannotti, DR, ISTI CNR Pisa	Rapporteur
Isabelle Guérin-Lassous, Professeur, Université Claude Bernard Lyon 1	Examinateur
Katharina Morik, Professeur, Technical University of Dortmund	Rapporteur
Marie-Christine Rousset, Professeur, Université Joseph Fourier	Examinateur
Céline Rouveirol, Professeur, Université Paris 13	Rapporteur
Christel Vrain, Professeur, Université d'Orléans	Examinateur

Acknowledgments

Je tiens à remercier ici toutes les personnes qui m'ont donné la chance de travailler avec elles : Jean-François pour sa confiance, son soutient et les beaux projets dans lesquels il m'a impliquée; les plus jeunes de l'équipe – Jérémy, Ruggero, Loïc – dont le départ a souvent laissé un vide; mes pairs – Marc, Mehdi, Marian, Christophe – pour le cadre et le contenu du travail, extrêmement agréable et vivifiant; ceux un peu plus loin – Pierre, Patrick, Patrice, Eric, Marie, Luc – pour le plaisir que j'ai à travailler avec eux; ceux encore plus loin – Elisa, Bart, Toon, Boris, Adriana – avec qui j'ai partagé l'exotisme belge; enfin les jeunes encore là, dans leurs premiers apprentissages du métier de chercheur – Elise, Ronan – travailler avec vous est un plaisir.

Je veux aussi remercier chaleureusement, mes rapportrices – Fosca Giannotti, Katharina Morik et Céline Rouveirol – pour s'être plongé dans ce mémoire et m'avoir fait part de leurs commentaires; ainsi que mes examinatrices – Isabelle Guérin-Lassous, Marie-Christine Rousset et Christel Vrain – pour m'avoir fait l'honneur d'accepter de donner leur avis sur mon travail.

Et puis, il y a ceux dont la vie est un peu bousculée parfois par mon travail – Antoine, Antonin et Gaspard et la famille – merci pour tout!!

Contents

Introduction

Knowledge discovery in databases (KDD) aims to assist humans in making sense of voluminous data. It consists in embedding raw data into predictive or descriptive models to elicit hypothesis on the mechanisms that generate the data. The whole process is generally decomposed in five steps (Fayyad, Piatetsky-shapiro, and Smyth 1996) that are: (1) the selection of a subset of the data samples and descriptors in adequacy with the targeted goal of the user; (2) the preprocessing of the data by handling the possible missing data, the errors of measurement or the outliers; (3) the transformation of the data by aggregating the initial descriptors into useful features or discretizing the descriptor values; (4) the data mining process that produces a particular enumeration of patterns or models over the data and (5) the interpretation and evaluation of the output of the previous step. Our contributions concern the data mining step and are all related to the descriptive paradigm that focuses on finding human-interpretable patterns that characterize the data and provide new and actionable insights. This unsupervised exploration of the data aims to highlight the relationships or structures that link objects, descriptors and objects to descriptors.

Data mining algorithms are defined by the language used to describe discoverable patterns or models, the criteria applied to evaluate the suitability of a particular pattern for the data, and the search strategy employed to retrieve the patterns that satisfy some conditions on the evaluation criteria. In my first contribution (Robardet 2002), I designed a co-clustering method, that, as any other clustering approaches, seeks to group together similar objects, but also, simultaneously clusters the object descriptors, exploiting the duality between objects and descriptors. It provides a model, that is to say a high level global description of the data. The language used is the set of co-clusterings, i.e. a pair made of a partition of the object set and another one of the descriptor set. The adjustment of one partition to the other is evaluated based on the Goodman-Kruskal's τ, a measure for cross-association in contingency tables that quantifies the strength of the relationship between two categorical variables. Due to the fact that this measure has a defined upper limit that is independent of the numbers of clusters, the τ function can be used to compare co-clusterings with different number of clusters. The co-clustering of any size that maximizes τ is approximated by a local search strategy. This work has been recently extended to handle star-structured data made of different views that are as many distinct descriptions of the same set of objects (Ienco, Robardet, R. G. Pensa, and Meo 2013).

Joining Jean-François Boulicaut research team in 2001, I started to work on the *constraint-based pattern mining framework* which covers data mining algorithms that use an exact search

strategy to extract the whole set of patterns satisfying some constraints on the evaluation criteria. At that time, the only considered patterns were local, that is to say patterns whose validity is evaluated independently from other patterns. Since then, this framework has been extended to the study of the active use of constraints on sets of patterns (Crémilleux and Soulet 2008; Raedt and Zimmermann 2007), in order to provide the correct answer to an inductive query: A query on both data and patterns implicitly present in them. Upon my arrival in the research team, I have been immediately involved in the cInQ European project[1] and its follow-up IQ[2] whose objectives were to study this framework.

In the early 2000's, frequent itemsets, association rules and episodes were the pattern types the most studied within this framework. Frequent itemsets are sets of Boolean descriptors that are simultaneously satisfied by a large number of objects. During Jérémy Besson's PhD, we proposed to exploit the duality between objects and descriptors by considering these patterns as bi-sets made of a set of descriptors (or itemset) associated to a set of objects supporting them. On this language of patterns, we defined constraints that can be applied equally to one set or the other. In particular, we studied the constraints to extract formal concepts (Ganter and Wille 1997). We designed an algorithm that is completely symmetrical and can be used even on dense data sets with no small dimension (Besson, Robardet, and Boulicaut 2004; Besson, Robardet, Boulicaut, and Rome 2005). This work has led to further developments in two main directions: (1) the extraction of fault-tolerant patterns (Besson, Robardet, and Boulicaut 2006) to be more robust to the possible errors of measurement in real-life data, and (2) its extension to the extraction of patterns in n-ary relations (Cerf, Besson, Robardet, and Boulicaut 2008; Cerf, Besson, Robardet, and Boulicaut 2009), all along Loïc Cerf's PhD.

Ruggero Pensa's PhD, that I co-supervised with Jean-François Boulicaut, combines the two aforementioned contributions in a generic framework (R. G. Pensa, Robardet, and Boulicaut 2005) to compute a co-clustering from collections of local patterns. Local patterns make possible to capture locally strong associations between objects and descriptors, but their use is limited by the huge size of the obtained collections that makes difficult their interpretation and validation by end-users. Building a co-clustering on top of these local patterns provides a good and robust summary of the data. Constraints on the co-clustering specified by end-users (e.g., must-link, cannot-link constraints between objects or descriptors) can be exploited during the local pattern extraction as well as during the co-clustering construction. Doing so, it improves the performance of the search strategies used in both steps.

During the first half of 2008, I received a sabbatical and went to Antwerp (Belgium) in ADReM (Advanced Database Research and Modelling) research team led by professor Jan Paredaens. I worked with Bart Goethals and Boris Cule on sequence pattern mining (Cule, Goethals, and Robardet 2009). To extract knowledge from event sequences, we proposed a new interestingness measure that evaluates the cohesion and the frequency of a pattern. We adapted the algorithmic principles identified for the extraction of bi-sets to the extraction of set patterns in sequences. They reduce, as soon as possible, the search space based on an upper-bound of the constraint making the algorithm very efficient. I also worked on an inductive database prototype called MININGVIEWS (Blockeel, Calders, Fromont, Goethals, Prado, and Robardet 2012), developed by Adriana Prado in her PhD. MININGVIEWS is an inductive database that, as such, integrates database querying with database mining. Its particularity is to use a plain SQL query language to express the mining queries. This intuitive and elegant framework is based on virtual mining views, which are relational tables that virtually contain the complete output of data mining algorithms executed over a given data table. Several types of patterns and models, such as itemsets, association rules, and decision trees, can be represented and queried with SQL

---

[1] Consortium on knowledge discovery using Inductive Queries (cInQ) (IST FET 2000-26469), 2001-2004

[2] Inductive Queries (IQ) (IST FET FP6-516169), 2005-2008.

using a unifying framework. I worked more specifically on a complete data mining scenario with SQL queries over the mining views. This proof of concepts illustrates the potential of the framework and was executed in MININGVIEWS system during a demonstration at SIGKDD (Blockeel, Calders, Fromont, Goethals, Prado, and Robardet 2008).

The Rhône Alpes Complex Systems Institute IXXI offered me the opportunity to create new collaborations with researchers on graph theory and signal processing and work with them on real-life complex networks. I obtained in 2008-2009 an INRIA delegate position in the D-Net team led by Éric Fleury. In this context, I worked on two analysis of dynamic graphs. In the first one, we proposed a framework for the study of dynamic mobility networks that addresses the description, analysis and the simulation of wireless sensor mobility networks (Scherrer, Borgnat, Fleury, Guillaume, and Robardet 2008) using techniques from signal processing, graph theory and data mining. The second research axis focuses on the study of the Vélo'v bicycle sharing system of Lyon viewed as a dynamic complex network. We conducted several analysis: In Jensen, Rouquier, Ovtracht, and Robardet 2010, we showed that bicycles compete with cars in term of speed; We proposed in Borgnat, Abry, Flandrin, Robardet, Rouquier, and Fleury 2011 a global analysis of the activity in time for which a predictive model is developed using signal processing tools; We presented in Borgnat, Robardet, Abry, Flandrin, Rouquier, and Tremblay 2013 a joint analysis in space and time of the Vélo'v network based on the adaptation to the case of dynamical networks of community detection methods. In his in progress PhD, that I co-supervised with Patrick Flandrin and Pierre Borgnat, Ronan Hamon studies the Vélo'v system, performing a frequency analysis on the signals obtained by the transformation of the graph using classical multidimensional scaling (CMDS) (Hamon, Borgnat, Flandrin, and Robardet 2013b).

Considering the constraint-based pattern mining framework for the analysis of relational graphs, I contributed to the specification of three pattern types that are tailored to extract knowledge from attributed graphs, dynamic graphs or attributed dynamic graphs. In Prado, Plantevit, Robardet, and Boulicaut 2013, we proposed to mine the graph topology of a large static attributed graph by finding regularities among vertex descriptors that are either the vertex attributes or topological properties used to describe the connectivity of each vertex in the graph. These descriptors are of numerical or ordinal types and their similarity is captured by quantifying their co-variation, that is, if their largest or smallest values are supported mostly by the same set of vertices. We proposed several interestingness measures and designed an efficient algorithm that combines searching and pruning strategies in the identification of the most relevant topological patterns. To probe relationships in real-life systems that are mostly dynamic, with vertices and edges appearing or disappearing through time, I proposed in Robardet 2009 to look for temporal interactions in dynamic relational graphs. The extracted evolving patterns are highly connected sub-graphs that undergo temporal modifications from one time stamp to the next. These subgraphs can grow, shrink, be stable, disappear or appear over time. The method first computes subgraphs that capture locally strong associations between vertices, and then uses these local patterns to construct a global model of the graph's dynamics. The in progress PhD of Elise Desmier focuses on the analysis of attributed dynamic graphs. In Desmier, Plantevit, Robardet, and Boulicaut 2012 and Desmier, Plantevit, Robardet, and Boulicaut 2013, the constraint-based pattern mining framework is used to identify dynamic sub-graphs in attributed dynamic graphs. In these patterns, the subgraph vertices follow the same temporal trends over a subset of attributes. This pattern domain relies on the graph structure and the temporal evolution of the attribute values. Several interestingness measures are proposed to retrieve the most relevant patterns with regard to the graph structure, the vertex attributes, and the time.

This *Habilitation à Diriger des Recherches* thesis is organized as follows. In the first chapter, I present the constraint-based pattern mining framework, exhibit the constraints properties that have been identified as useful either to reduce the number of irrelevant patterns or to truncate the

search space while preserving the completeness of the extraction. The special case of formal concept extraction under constraints on objects and descriptors is then discussed. After that, I introduce the principles of a generic algorithmic that pushes monotone, anti-monotone and piecewise monotone constraints. The instantiation of this generic algorithm on two particular problems, that are the extraction of closed patterns in n-ary relations and the mining of parallel episodes in sequences, ends this chapter.

The second chapter presents two case studies of mobility networks. In the first one, the analysis focuses on sensor wireless mobility networks, where sensors were attached to people and measured their proximity to other people. To describe these networks, some graph properties, viewed as function of time, are studied so as to give an empirical statistical characterization of the dynamics. Global indicators from the dynamics of the network are also computed that are measured over several time steps. Then, several generic random dynamic models are proposed that make possible to generate random dynamic graphs which have a behavior similar to the one observed in experimental data sets. Their ability to simulate credible data is assessed by the similarity of the descriptors on both original and simulated data. The second case study is the bicycle sharing system of Lyon, called Vélo'v. I present an analysis on the spatial and temporal aspects of this dynamical network. Community detection mining tools are used to extract either communities of stations that exchange regularly a large number of bicycles, or communities of stations that are similar in the time patterns of bicycle flows. These analyzes provide a significant understanding of the social use of the Vélo'v program in Lyon.

The third chapter discusses my main contributions on local pattern discovery in attributed and/or dynamic relational graphs. I first present an approach to characterize the relationship between vertex attributes and the graph topology in static attributed graphs. It consists in the extraction of co-variations between vertex attributes and measures describing the relationship of the vertex with the rest of the graph. Then, I propose to analyze dynamic graphs by discovering the main temporal changes as locally strong associations between vertices and their evolution through time. Finally, I introduce the mining of trends in attributed dynamic graphs to identify connected parts of the graph whose vertex attributes evolve in the same way.

The last chapter concludes this report and sketches some future works.

1 — Constraint-based pattern mining
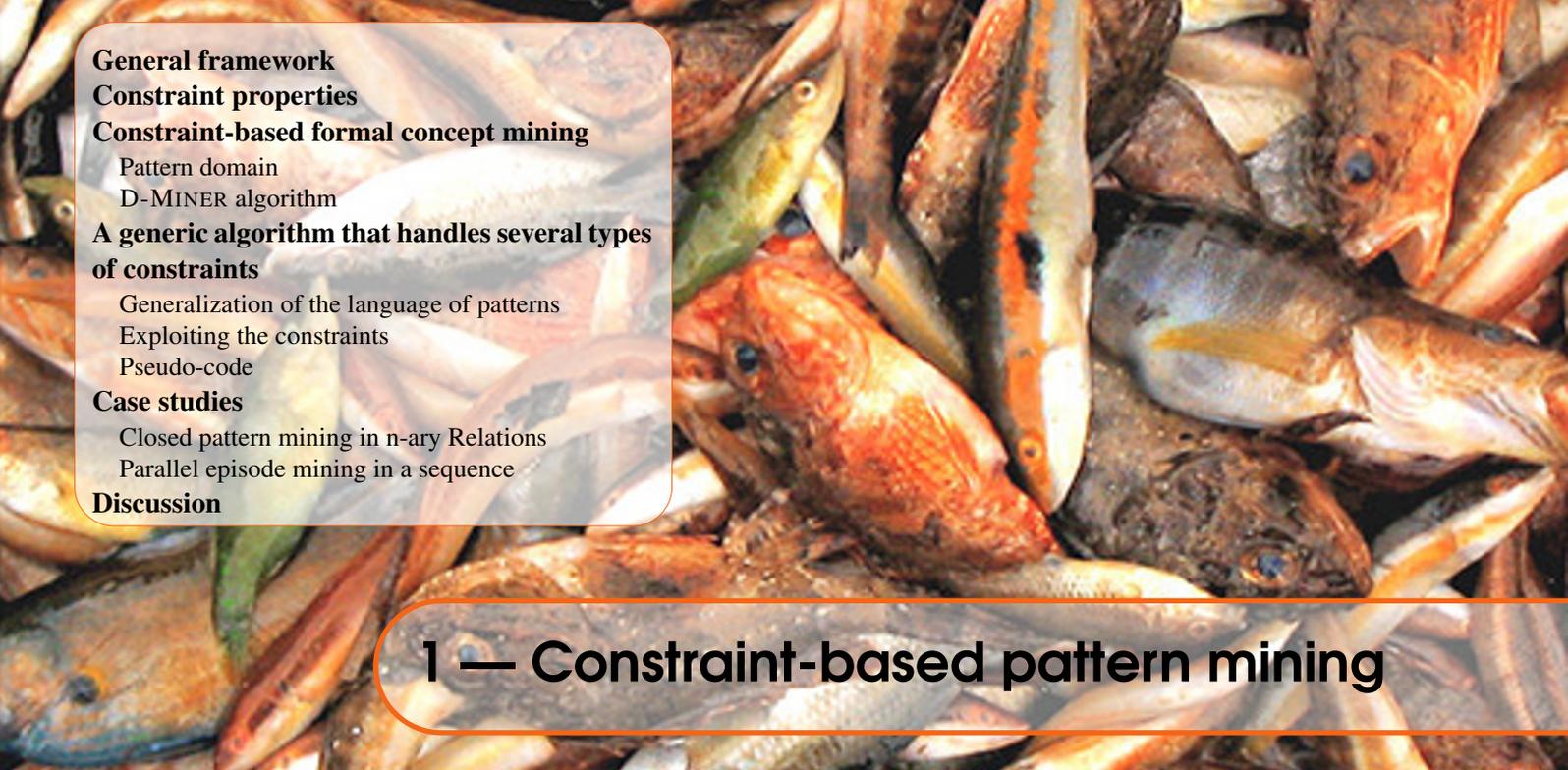
1.1 General framework

Constraint-based pattern mining covers data mining algorithms that use an exact search strategy to achieve the exhaustive extraction of the whole set of patterns satisfying some constraints over the data. These constraints are Boolean expressions based on evaluation criteria that measure the relevance of patterns in a specific data set. As the languages of patterns are of exponential size, it is impossible to list all the patterns and verify the constraints on them all. Therefore, constraints are needed for two reasons. First, the constraints make it possible to discover knowledge in the data by reducing the number of patterns that are presented to the user while retrieving those that may best summarize the highlights of the data. The constraints diminish the number of irrelevant patterns and constitute as such a solution to the pattern flooding problem. Second, they increase the computational efficiency of the process. More precisely, they make the extraction feasible by reducing drastically the number of patterns on which the constraints must be evaluated. They can be pushed deep inside the mining algorithm in order to truncate the search space while preserving the completeness of the extraction. This framework requires the specification of a language of patterns and of a set of constraints to compute all the patterns that satisfy the constraints in the data. Formally, it is defined (Mannila and Toivonen 1997) as the discovery of the theory $\text{Th}(\mathscr{L}, \mathscr{D}, \mathscr{C}) = \{\varphi \in \mathscr{L} \mid \mathscr{C}(\varphi, \mathscr{D}) \text{ is true}\}$ where $\mathscr{D}$ is the data to be mined, $\mathscr{L}$ is the language of patterns, and $\mathscr{C}(\varphi, \mathscr{D})$ is a constraint that states whether the pattern φ fit the data $\mathscr{D}$. The constraint $\mathscr{C}$ is specified by the user to drive the mining process towards what is potentially interesting for the current application.

Thus, a constraint is particularly interesting if (1) it makes possible to properly assess the quality of patterns, and (2) has properties that, when combined with a smart enumeration process, provides the means to infer its value on many patterns without having to evaluate them individually. Several useful constraint properties have been identified so far and designing fast, scalable and generic algorithm requires to be able to simultaneously handle a large variety of constraints. Such algorithms may implement pruning strategies and propagation mechanisms for as much as possible types of constraints.

In the following section, I present the constraint properties that have been identified as useful in constraint-based pattern mining. Then, section 1.3 presents D-MINER, an algorithm that extracts formal concepts using monotone constraints. In section 1.4, I present a generic algorithm that has capabilities to push monotone, anti-monotone and piecewise monotone constraints.

Finally, section 1.5 reviews two contributions on the extraction of specific pattern types.

1.2 Constraint properties

Constraint properties are defined up to a partial order relation between patterns. Therefore, before presenting the different constraint properties and their computational use, we introduce the notation $\preceq$, which refers to the partial order used when listing the patterns.

> **Definition 1.1 — Partial order on patterns.** Let $\preceq$ be a partial order over $\mathscr{L}$, that is to say a binary relation that is reflexive, anti-symmetric and transitive. Given two patterns φ_1 and φ_2 in $\mathscr{L}$, φ_1 is said to be more specific than φ_2 iff $\varphi_1 \preceq \varphi_2$.

The examples given in the following assumes that $\mathscr{L}$ is the language of set patterns that are partially ordered by $\subseteq$.

Constraints that have monotone or anti-monotone property can be exploited to prune large part of the search space.

> **Definition 1.2 — Monotone and anti-monotone constraints.** Let $\varphi_1, \varphi_2 \in \mathscr{L}$ such that $\varphi_1 \preceq \varphi_2$. A constraint $\mathscr{C}$ is monotone if and only if $\mathscr{C}(\varphi_1, \mathscr{D}) \Rightarrow \mathscr{C}(\varphi_2, \mathscr{D})$. $\mathscr{C}$ is anti-monotone if and only if $\mathscr{C}(\varphi_2, \mathscr{D}) \Rightarrow \mathscr{C}(\varphi_1, \mathscr{D})$.

Constraints that are either monotone or anti-monotone are the ones that are easiest to exploit in constraint-based pattern mining. If a pattern φ_2 does not satisfy a monotone constraint $\mathscr{C}$, then all the patterns φ_1 such that $\varphi_1 \preceq \varphi_2$ can be pruned. If a pattern φ_1 does not satisfy an anti-monotone constraint $\mathscr{C}$, then all the patterns φ_2 such that $\varphi_1 \preceq \varphi_2$ can be pruned.

■ **Example 1.1** Let v be a positive real-valued attribute associated to the elements of a pattern φ and let $\mathscr{C}(\varphi, v) \equiv \text{sum}(v(\varphi)) \leq \sigma$, which states that the sum of φ on v should be smaller than σ, with $\text{sum}(v(\varphi)) = \sum_{e \in \varphi} v(e)$. This constraint is anti-monotone: if φ does not satisfy the constraint then adding an element e to φ and the positive value $v(e)$ to its sum cannot lead to the satisfaction of the constraint. ■

Any conjunction of monotone (resp. anti-monotone) constraints is a monotone (resp. anti-monotone) constraint. As a consequence, any conjunction of monotone and anti-monotone constraints can be reduced to $\mathscr{C}_{monotone} \wedge \mathscr{C}_{anti-monotone}$.

Mining algorithms can also profit from constraints known to be succinct. A constraint satisfying this property can be translated into another constraint that has to be satisfied by each element of the pattern:

> **Definition 1.3 — Succinct constraints.** A constraint $\mathscr{C}$ is succinct if and only if there exists a constraint $\mathscr{S}$ such that for all pattern $\varphi \in \mathscr{L}$, $\mathscr{C}(\varphi, \mathscr{D}) \equiv \forall e \in \varphi$, $\mathscr{S}(e, \mathscr{D}) = true$.

One can test whether a pattern satisfies a succinct constraint by testing that all its members satisfy the related constraint. Such constraint can generally be exploited in a pre-processing step by restraining the language of patterns to the ones with elements satisfying $\mathscr{S}$.

■ **Example 1.2** Let v be a real-valued attribute associated to the elements of a pattern φ and let $\mathscr{C}(\varphi, v) \equiv \max(v(\varphi)) < \sigma$, which states that the maximum value of φ on v should be smaller than σ. Thus, $\mathscr{C}(\varphi, v) \equiv \forall e \in \varphi$, $v(e) < \sigma$. Not enumerating the elements having a value on v greater than σ leads to the constraint satisfaction. ■

Convertible constraints, introduced in Pei and Han 2000, are constraints for which there does not exist a reciprocal relationship between the partial order $\preceq$ on patterns and the constraint satisfaction. However, such an interplay can be obtained by specifying a prefix order $\leq$ over the patterns. Such a prefix order is a partial order that is downward total: $\forall \varphi_1 \preceq \varphi_2$, then either $\varphi_1 \leq \varphi_2$ or $\varphi_2 \leq \varphi_1$.

> **Definition 1.4 — Convertible constraints.** A constraint $\mathscr{C}$ is convertible anti-monotone if (1) the partial order $\preceq$ is extended to a prefix order $\leq$ on patterns of $\mathscr{L}$ and (2) whenever φ_2 satisfies $\mathscr{C}$, so does any pattern φ_1 such that $\varphi_1 \leq \varphi_2$.
>
> Similarly, $\mathscr{C}$ is convertible monotone iff $\forall \varphi_1, \varphi_2 \in \mathscr{L}$ such that $\varphi_1 \leq \varphi_2$, $\mathscr{C}(\varphi_1, \mathscr{D}) \Rightarrow \mathscr{C}(\varphi_2, \mathscr{D})$.

Handling such constraint requires to completely specify the enumeration order of the patterns of $\mathscr{L}$: Patterns are enumerated according to $\leq$ in order to make the constraint monotone (resp. anti-monotone) on every branch of the enumeration tree. Thus, the same pruning techniques, as those used for monotone (resp. anti-monotone) constraints can be used. However, it is impossible to take advantage of several convertible constraints in a single extraction process.

■ **Example 1.3** Let v be a real-valued attribute associated to the elements of a pattern φ and let $\mathscr{C}(\varphi, v) \equiv \mathrm{avg}(v(\varphi)) > \sigma$, which states that the average value of φ on v should be larger than σ. Listing the elements of φ in descending order of their value on v makes the constraint anti-monotone.   ■

Bonchi and Lucchese 2007 defines a loose anti-monotone constraint as such that, if it is satisfied by a pattern of cardinality k then it is satisfied by at least one of its subsets of cardinality $k-1$:

> **Definition 1.5 — Loose anti-monotone constraints.** Given a pattern φ, a constraint $\mathscr{C}$ is loose anti- monotone if: $\mathscr{C}(\varphi, \mathscr{D}) \Rightarrow \exists e \in \varphi : \mathscr{C}(\varphi \setminus \{e\}, \mathscr{D})$.

■ **Example 1.4** Let v be a real-valued attribute associated to the elements of a pattern φ and let $\mathscr{C}(\varphi, v) \equiv \mathrm{var}(v(\varphi)) \leq \sigma$, which states that the variance of φ on v should be smaller than σ, with $\mathrm{var}(v(\varphi)) = \frac{\sum_{e \in \varphi}(v(e) - \mathrm{avg}(v(\varphi)))^2}{|\varphi|}$. If φ satisfies the constraint, so does $\varphi \setminus \{e\}$, where e is the element of φ whose value on v is the most far away from $\mathrm{avg}(v(\varphi))$, that is to say $\arg \max_{e \in \varphi}(v(e) - \mathrm{avg}(v(\varphi)))^2$.   ■

Loose anti-monotone constraints can be evaluated directly into a level-wise breadth-first exploration of the search space, performed by Apriori-like algorithms. In this algorithm, patterns of cardinality k are generated from patterns of cardinality $k-1$. Therefore, the set of valid patterns of cardinality $k-1$ is completely available when considering a pattern of size k and the constraint can be checked. However, breadth-first traversal of the search space is generally memory-space consuming which can incline to prefer a depth-first traversal of the search space.

Reverse search algorithms (Avis and Fukuda 1996; Georgii, Tsuda, and Schölkopf 2011; Uno 2010) are depth-first search algorithms suited to handle such constraints. In the reverse search approach, the search tree is specified by defining a reduction map $m(\varphi)$, which transforms a pattern φ into a unique pattern $\varphi' = \varphi \setminus \{e\}$ that satisfies the constraint. This pattern is the parent of φ in the enumeration tree. This makes the constraint anti-monotone on the search tree induced by the reduction map: Any pattern that satisfies the constraint descends from a pattern that also satisfies the constraint. To enumerate all the patterns that satisfy the anti-monotone constraint, one has to traverse the implicitly defined search tree in a depth-first manner. During the traversal, children are generated on demand if the constraint is satisfied. As the reduction

map defines how to get from children to parents and not vice versa, children cannot be directly derived from a given parent. Instead, to generate the children of a pattern φ, all candidates $\varphi \cup \{e\}$, $e \notin \varphi$ have to be considered. The actual children are the ones such that $m(\varphi \cup \{e\}) = \varphi$ (reverse search principle).

■ **Example 1.5**   Considering the constraint $\mathscr{C}(\varphi, v) \equiv \text{var}(v(\varphi)) \leq \sigma$ presented in Example 1.4, the reduction map is $m(\varphi) = m(\varphi \setminus \{u\})$ with $u = \arg \max_{e \in \varphi} (v(e) - \text{avg}(v(\varphi)))^2$. If several elements satisfy this equation, the one of smallest index is taken.　　　　■

Since $\mathscr{C}$ is now anti-monotone of the search tree, the enumeration can be stopped as soon as the constraint is not satisfied.

More generally, if one is capable to derive a bound with monotone property from a constraint that relies on a non-monotone measure, we say that the constraint is boundable. The idea of pruning with the information of upper bounds is a standard technique in combinatorial optimization than can also be used in pattern set mining (Morishita and Sese 2000).

> **Definition 1.6 — Boundable constraints.**   A constraint $\mathscr{C}(\varphi, \mathscr{D}) \equiv g(\varphi) \geq \sigma$, with $g : \mathscr{L} \to \mathbb{R}$, is said to be upper-boundable if one can derive a function g' such that $g(\varphi_2) \leq g'(\varphi_1)$ for all $\varphi_1 \preceq \varphi_2$. Therefore, if $g'(\varphi_1) < \sigma$ all patterns φ_2 can be pruned.
> Similarly, $\mathscr{C}(\varphi, \mathscr{D}) \equiv g(\varphi) \leq \sigma$ is said to be lower-boundable if one can derive a function g' such that $g(\varphi_2) \geq g'(\varphi_1)$ for all $\varphi_1 \preceq \varphi_2$. Therefore, if $g'(\varphi_1) > \sigma$ all patterns φ_2 can be pruned.

An example of such constraint is presented in Section 3.1.1. However, from this definition, we cannot infer a systematic method for deriving a bound from any boundable constraint.

On the contrary, in Cerf, Besson, Robardet, and Boulicaut 2008; Cerf, Besson, Robardet, and Boulicaut 2009, we identify a new class of constraints, the piecewise monotone and anti-monotone constraints, for which a bound can be systematically inferred.

> **Definition 1.7 — Piecewise monotone and anti-monotone constraints.**   Let $\mathscr{C}$ be a compound constraint whose expression involves p times the pattern φ. $\mathscr{C}$ can be reformulated as $\mathscr{C}(\varphi, \mathscr{D}) = f(\varphi_1, \cdots \varphi_p, \mathscr{D})$. Thus, we can study the partial monotony of the constraint by considering the function $f_{i,\varphi}(x) = (\varphi_1, \cdots \varphi_{i-1}, x, \varphi_{i+1}, \cdots, \varphi_p, \mathscr{D})$. $\mathscr{C}$ is piecewise monotone and anti-monotone iff $\forall i = 1 \ldots p$, $f_{i,\varphi}$ is either monotone or anti-monotone:
>
> $$\forall x, y \in \mathscr{L} \text{ such that } x \preceq y, \begin{cases} f_{i,\varphi}(x) \Rightarrow f_{i,\varphi}(y) \text{ iff } f_{i,\varphi} \text{ is monotone} \\ f_{i,\varphi}(y) \Rightarrow f_{i,\varphi}(x) \text{ iff } f_{i,\varphi} \text{ is anti-monotone} \end{cases}$$

■ **Example 1.6**   Let v be a positive real-valued attribute associated to the elements of a pattern φ, $\mathscr{C}(\varphi, \mathscr{D}) \equiv \text{avg}(v(\varphi)) > \sigma$, and $\text{avg}(v(\varphi)) = \frac{\sum_{e \in \varphi} v(e)}{|\varphi|}$. $\mathscr{C}(\varphi, \mathscr{D})$ is piecewise monotone and anti-monotone with $\mathscr{C}(\varphi, \mathscr{D}) = f(\varphi_1, \varphi_2, \mathscr{D})$ and

$$f(\varphi_1, \varphi_2, \mathscr{D}) = \frac{\sum_{e \in \varphi_1} v(e)}{|\varphi_2|}$$

- $f_{1,\varphi}$ is monotone: $\forall x \preceq y$, $f(x, \varphi_2, \mathscr{D}) = \frac{\sum_{e \in x} v(e)}{|\varphi_2|} > \sigma \Rightarrow \frac{\sum_{e \in y} v(e)}{|\varphi_2|} > \sigma$
- $f_{2,\varphi}$ is anti-monotone: $\forall x \preceq y$, $f(\varphi_1, y, \mathscr{D}) = \frac{\sum_{e \in \varphi_1} v(e)}{|y|} > \sigma \Rightarrow \frac{\sum_{e \in \varphi_1} v(e)}{|x|} > \sigma$.

　　　　■

This definition provides an operational way to handle this type of constraint on condition that the mining algorithm has capabilities to exploit both monotone and anti-monotone constraints.

The way these constraints are pushed into such a generic algorithm presented in Section 1.4 is detailed in Section 1.4.2. Note that piecewise monotone and anti-monotone constraints are similar to the primitive-based constraints defined in Soulet and Crémilleux 2006.

1.3 Constraint-based formal concept mining

1.3.1 Pattern domain

The first mining task we worked on is that of formal concepts extraction (Besson, Robardet, and Boulicaut 2004; Besson, Robardet, Boulicaut, and Rome 2005). Considering data tables that represent binary relations between objects and properties, a formal concept is a pair (O,P) composed of a set of objects O and a set of properties P such that objects in O have all the properties from P, and vise-versa. Besides, the pair is maximal: any object, that does not belong to O, does not satisfy a property from P, and any property, that does not belong to P is not satisfied by at least an object from O.

More formally, let $\mathscr{O}$ be a set of objects or transactions and $\mathscr{P}$ a set of properties or items. The data to be mined are the relation $\mathscr{D} \subseteq \mathscr{O} \times \mathscr{P}$. Figure 1.1 provides an example of such data set where $\mathscr{O} = \{o_1, \ldots, o_5\}$ and $\mathscr{P} = \{p_1, \ldots, p_{10}\}$. $(o_i, p_j) \in \mathscr{D}$ denotes that property p_j holds for objects o_i.

On such data, we are looking for bi-sets, that is a set of objects that is associated to a set of properties:

> **Definition 1.8 — Language of bi-sets.** The language of bi-sets $\mathscr{L}$ is the set of pairs (O,P) such that $O \subseteq \mathscr{O}, P \subseteq \mathscr{P}$.

					Properties					
	p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}
o_1	1	1	1	1	0	1	1	0	0	0
o_2	1	1	1	1	0	0	0	0	1	1
o_3	1	1	1	1	0	0	0	0	1	1
o_4	0	0	0	0	1	1	1	1	1	1
o_5	1	0	1	0	1	1	1	1	0	0

Figure 1.1: Example of a data set $\mathscr{D}_1$

Formal concepts are specific bi-sets that satisfy constraints defined thanks to derivation operators.

> **Definition 1.9 — Derivation operators.** For a set of object $O \subseteq \mathscr{O}$, we define the set of properties that all objects in O have in common as follows:
>
> $$\phi(O, \mathscr{D}) = \{p \in \mathscr{P} \mid \forall o \in O, (o,p) \in \mathscr{D}\}$$
>
> In a dual way, for a set of properties $P \subseteq \mathscr{P}$, we define the set of objects that have all properties from P as follows:
>
> $$\psi(P, \mathscr{D}) = \{o \in \mathscr{O} \mid \forall p \in P, (o,p) \in \mathscr{D}\}$$

■ **Example 1.7** In $\mathscr{D}_1$ from Figure 1.1, $\phi(\{o_3, o_4\}, \mathscr{D}_1) = \{p_9, p_{10}\}$ and $\psi(\{p_1, p_2\}, \mathscr{D}_1) = \{o_1, o_2, o_3\}$. ■

> **Definition 1.10 — Formal concept.** A formal concept of $\mathscr{D}$ is a bi-set (O, P) such that $\phi(O, \mathscr{D}) = P$ and $\psi(P, \mathscr{D}) = O$.

■ **Example 1.8** In $\mathscr{D}_1$ from Figure 1.1, there are twelve formal concepts including $(\{o_2, o_3\}, \{p_1, p_2, p_3, p_4, p_9, p_{10}\})$ and $(\{o_1, o_2, o_3, o_5\}, \{p_1, p_3\})$. ■

D-MINER computes the language of formal concepts under monotone constraints. Its enumeration process relies on the following partial order on bi-sets.

> **Definition 1.11 — Partial order on bi-sets.** Let (O_1, P_1) and (O_2, P_2) be two bi-sets. $(O_1, P_1) \preceq (O_2, P_2)$ iff $O_1 \subseteq O_2$ and $P_1 \subseteq P_2$.

The monotone constraints that are implemented in D-MINER are the minimal size constraints together with syntactical constraints on both sets O and P:

- The monotone constraint on O pushed in the algorithm is $\mathscr{C}_{\mathscr{O}}((O, P), \mathscr{D}) \equiv |O| \geq \sigma_{\mathscr{O}} \wedge S_{\mathscr{O}} \subseteq O$, with $\sigma_{\mathscr{O}}$ a real value that specifies the minimal number of elements that O must contain, and $S_{\mathscr{O}} \subseteq \mathscr{O}$ is a set of objects that has to belong to O.
- In a dual way, $\mathscr{C}_{\mathscr{P}}((O, P), \mathscr{D}) \equiv |P| \geq \sigma_{\mathscr{P}} \wedge S_{\mathscr{P}} \subseteq P$.

1.3.2 D-MINER algorithm

A formal concept (O, P) is a maximal bi-set such that all its properties and objects are in relation by $\mathscr{D}$. Thus, the absence of relation between an object o and a property p generates two formal concepts: One that contains p but not o, and another one that contains o but not p. To be able to quickly remove objects and properties that are not in relation, D-MINER uses a set H of all pairs (o, Q) with $o \in \mathscr{O}$, $Q = \mathscr{P} \setminus \phi(\{o\}, \mathscr{D})$. Such elements are called *"cutters"* and are bi-sets made of an object o and the set of properties not in relation with o in $\mathscr{D}$.

■ **Example 1.9** The set H of cutters used to mine $\mathscr{D}_1$ (see Figure 1.1) is made of $(\{o_1\}, \{p_5, p_8, p_9, p_{10}\})$, $(\{o_2\}, \{p_5, p_6, p_7, p_8\})$, $(\{o_3\}, \{p_5, p_6, p_7, p_8\})$, $(\{o_4\}, \{p_1, p_2, p_3, p_4\})$ and $(\{o_5\}, \{p_2, p_4, p_9, p_{10}\})$. ■

Starting from the pair $(\mathscr{O}, \mathscr{P})$, D-MINER recursively splits the current pair using the elements of H. If all the elements of H have been used (line 1 of the function), the current pair (X, Y) is a formal concept that is added to the output. Otherwise, (o, Q), the i^{st} element of H, is considered (line 4). If (o, Q) cannot be applied to (X, Y) because there is no intersections between (o, Q) and (X, Y), that is to say $o \notin X$ or $P \cap Y = \emptyset$, the next cutter is considered thanks to a recursive call (line 6). Otherwise, (o, Q) is used to cut (X, Y):

- If the monotone constraint $\mathscr{C}_{\mathscr{O}}$ is satisfied (line 8), o is removed from X since the properties of $Y \cap Q$ are not satisfied by o. (o, Q) is added to H_L to further check that $Y \cap Q$ will not get empty, otherwise (X, Y) might not be maximal. Then, the enumeration process continues with a recursive call of the function (line 9); Note that, as each object o belongs to one and only one element of H, an object o is at most removed once during the enumeration process.
- The process is a bite more tricky when considering the removal of properties, as a property may belong to several elements of H. So, similarly as what was done in the previous case, if the monotone constraint $\mathscr{C}_{\mathscr{P}}$ is satisfied (line 11), the properties of Q might be removed from Y, provided that the candidate $(X, Y \setminus Q)$ is still maximal. This is evaluated from line 12 to line 18: if there exists an element (o', Q') from H_L such that $(Y \setminus Q) \cap Q' =$ then $(X, Y \setminus Q)$ is not maximal as o' could be added to it. In that case the enumeration process stops. Otherwise Q is removed from Y and the enumeration continues (line 19).

Algorithm 1 D-MINER

Require: Relation $\mathscr{D}$ with $\mathscr{O}$ the set of objects, $\mathscr{P}$ the set of properties, $\mathscr{C}_{\mathscr{O}}$ and $\mathscr{C}_{\mathscr{P}}$ the monotone constraints on $\mathscr{O}$ and $\mathscr{P}$.

Ensure: $\mathscr{R}$ the set of all formal concepts that satisfy the constraints $\mathscr{C}_{\mathscr{O}}$ and $\mathscr{C}_{\mathscr{P}}$.

1: $H_L \leftarrow$ empty()
2: $H \leftarrow \{(o,Q) \mid o \in \mathscr{O} \text{ and } Q = \mathscr{P} \setminus \phi(\{o\},\mathscr{D})\}$
3: $\mathscr{R} \leftarrow$ D-Miner_Enumeration$((\mathscr{O},\mathscr{P}),H,H_L,0)$

Function D-Miner_Enumeration$((X,Y),H,H_L,i)$

Require: (X,Y) a couple of $2^{\mathscr{O}} \times 2^{\mathscr{P}}$, H the list of cutters, H_L the set of elements of H used to removed objects, i the depth of the recursion.

Ensure: $\mathscr{R}$ the set of concepts that contain (X,Y) and satisfy $\mathscr{C}_{\mathscr{O}}$ and $\mathscr{C}_{\mathscr{P}}$

1: **if** $i = |H|$ **then**
2:    **return** $\mathscr{R} \leftarrow \mathscr{R} \cup (X,Y)$
3: **else**
4:    $(o,Q) \leftarrow H[i]$
5:    **if** $((o \cap X = \emptyset)$ or $(Q \cap Y = \emptyset))$ **then**
6:      $\mathscr{R} \leftarrow \mathscr{R} \cup$ D-Miner_Enumeration$((X,Y),H,H_L,i+1)$
7:    **else**
8:      **if** $\mathscr{C}_{\mathscr{O}}((X \setminus \{o\},Y),\mathscr{D})$ is satisfied **then**
9:        $\mathscr{R} \leftarrow \mathscr{R} \cup$ D-Miner_Enumeration$((X \setminus \{o\},Y),H,H_L \cup (o,Q),i+1)$
10:      **end if**
11:      **if** $\mathscr{C}_{\mathscr{P}}((X,Y \setminus Q),\mathscr{D})$ is satisfied **then**
12:        $isClosed \leftarrow$ **true**
13:        **for all** $(o',Q') \in H_L$ **do**
14:          **if** $((Q' \cap Y) \setminus Q = \emptyset)$ **then**
15:            $isClosed \leftarrow$ **false**
16:          **end if**
17:        **end for**
18:        **if** $isClosed$ **then**
19:          $\mathscr{R} \leftarrow \mathscr{R} \cup$ D-Miner_Enumeration$((X,Y \setminus Q),H,H_L,i+1)$
20:        **end if**
21:      **end if**
22:    **end if**
23: **end if**
24: **return** $\mathscr{R}$

■ **Example 1.10** Let us consider again Figure 1.1 and suppose that $(X,Y) = (\{o_1,o_2,o_3\}, \{p_1,p_2,p_3,p_4,p_5,p_6,p_7\})$ and let first remove o_1 using $(\{o_1\},\{p_5,p_8,p_9,p_{10}\})$. In a recursive call, we will consider the cutter $(\{o_2\},\{p_5,p_6,p_7,p_8\})$. If we use it to remove the properties p_5, p_6 and p_7, we will have as current candidate the pattern $(\{o_2,o_3\},\{p_1,p_2,p_3,p_4\})$ that is not a formal concept, as o_1 satisfies the properties p_1, p_2, p_3 and p_4. ■

In D-MINER, sets are implemented using bisets that makes possible to quickly perform set operations using bitewise operations. Proofs of correctness and completeness of are given in (Besson, Robardet, Boulicaut, and Rome 2005). Jérémy Besson has proved in his PhD thesis (Besson 2005) that the complexity delay of D-MINER is in the worst case in $O(n^2m)$, where n is $|\mathscr{O}|$ and m is $|\mathscr{P}|$. The complexity delay of D-MINER is in average in $(n - \log(K) + 1)O(nm)$, with K the number of formal concepts of $\mathscr{D}$.

1.4 A generic algorithm that handles several types of constraints

The work we have carried out subsequently aimed to abstract the algorithmic principles of D-MINER in two directions: (1) generalize the language of patterns and (2) expand the type of constraints pushed in the algorithm to optimize its efficiency. Let us re-examine the properties of constraints as defined in section 1.2. There is no doubt that monotone and anti-monotone constraints have to be handled by a generic algorithm. Succinct constraints can be used in a pre-processing step by removing from the data the elements that do not satisfy the constraint. To be pushed, convertible constraints requires to completely specify the enumeration order of the patterns. This way of handling convertible constraints has two drawbacks:

- It is impossible to enforce several convertible constraints unless they rely on the same order of patterns (in other terms, the convertibility is not preserved by conjunction);
- It is not possible to use heuristics on the listing process to speed up the extraction.

For these reasons, we believe that in a generic algorithm it is not desirable to lock the order in which the patterns are listed. Loose anti-monotone constraints are pushed either by using a level-wise exploration of the search space, known to be memory-space consuming, or by employing reverse search algorithmic principles that also fix the order in which the patterns are listed. Therefore, convertible and loose anti-monotone constraints are not pushed into the generic algorithm. Regarding boundable constraints, there is no systematic way to derive a bound from such constraints. Therefore, it seems very difficult to use them in a generic algorithm. Finally, piecewise monotone and anti-monotone constraints constitute a general class of constraints that can be exploited and propagated in a systematic manner.

In the following, I present a generic algorithm that pushes monotone, anti-monotone and piecewise monotone constraints.

1.4.1 Generalization of the language of patterns

Considering the data $\mathscr{D} \subseteq \mathscr{D}^1 \times \cdots \times \mathscr{D}^d$ and a language of set patterns $\mathscr{L} \subseteq 2^{\mathscr{D}^1} \times \cdots \times 2^{\mathscr{D}^d}$, a natural partial order $\preceq$ of $\mathscr{L}$ relies on set inclusion: $\varphi_1 = (D_1^1, \ldots, D_1^d) \preceq \varphi_2 = (D_2^1, \ldots, D_2^d)$ iff $D_1^\ell \subseteq D_2^\ell, \forall \ell = 1 \ldots d$. This partial order forms a lattice: For any nonempty finite subset of patterns $\mathscr{F} = \{\varphi_i \in \mathscr{L}, i = 1 \ldots k\}$, $\mathscr{F}^\vee = \left(\bigcup_i \varphi_i\right) = \left(\cup_i D_i^1, \ldots, \cup_i D_i^d\right)$ and $\mathscr{F}^\wedge = \left(\bigcap_i \varphi_i\right) = \left(\cap_i D_i^1, \ldots, \cap_i D_i^d\right)$ are respectively the join and meet elements. The bounds of the lattice are $\top = \mathscr{L}^\vee$ and $\bot = \mathscr{L}^\wedge$.

All the patterns of the theory $\mathscr{R} = \mathrm{Th}(\mathscr{L}, \mathscr{D}, \mathscr{C})$ can be enumerated using a binary partition algorithm (Uno 2010). It consists in choosing an element $e \in \mathscr{D}^1 \times \cdots \times \mathscr{D}^d$ and divides $\mathscr{R}$ into two sets: $\mathscr{R}_{+e}$ and $\mathscr{R}_{-e}$. $\mathscr{R}_{+e}$ gathers all the patterns of $\mathscr{R}$ that contain e, and $\mathscr{R}_{-e}$ groups those that do not contain e. Therefore, the element e belongs to $\mathscr{R}_{+e}^\wedge$ and e does not belong to $\mathscr{R}_{-e}^\vee$. If $\mathscr{R}_{+e}$ (resp. $\mathscr{R}_{-e}$) is not empty and $\mathscr{R}_{+e}^\vee \neq \mathscr{R}_{+e}^\wedge$ (resp. $\mathscr{R}_{-e}^\vee \neq \mathscr{R}_{-e}^\wedge$), it is recursively divided by choosing another element in $\mathscr{R}_{+e}^\vee \setminus \mathscr{R}_{+e}^\wedge$ (resp. $\mathscr{R}_{-e}^\vee \setminus \mathscr{R}_{-e}^\wedge$). If $\mathscr{R}_{+e}^\vee = \mathscr{R}_{+e}^\wedge$ and $\mathscr{C}(\mathscr{R}_{+e}^\vee, \mathscr{D})$ (resp. $\mathscr{R}_{-e}^\vee = \mathscr{R}_{-e}^\wedge$ and $\mathscr{C}(\mathscr{R}_{-e}^\vee, \mathscr{D})$), then $\mathscr{R}_{+e}^\vee$ (resp. $\mathscr{R}_{-e}^\vee$) is a pattern of $\mathscr{R}$.

The number of iterations of a binary partition algorithm is linear in $|\mathscr{R}|$, which is the output size, if it is possible to check whether either $\mathscr{R}_{+e}$ or $\mathscr{R}_{-e}$ are empty in constant time. If this check can be done in polynomial time of the input data size, the binary partition algorithm is said to be polynomial delay. In the following, we explain how this test can be performed, depending on the constraint properties.

1.4.2 Exploiting the constraints

Monotone, anti-monotone and piecewise monotone constraints can be used in this generic algorithm to prune large parts of the search space. This is done thanks to two mechanisms: The verification and the propagation of constraints.

Constraint verification

Checking whether the search space is empty can be done by evaluating the constraints on the join or the meet elements of the search space. Indeed, if a monotone constraint is not satisfied by the join element $\mathscr{R}^\vee$, then $\mathscr{R}$ is empty. Similarly, if an anti-monotone constraint is not satisfied by the meet element $\mathscr{R}^\wedge$, then $\mathscr{R}$ is also empty.

> **Property 1.1 — Monotone constraint evaluation.** Let $\mathscr{C}$ be a monotone constraint. If $\mathscr{C}(\mathscr{R}^\vee, \mathscr{D})$ is not satisfied, then $\forall \varphi \in \mathscr{R}$, $\mathscr{C}(\varphi, \mathscr{D})$ is not satisfied.

Proof. $\forall \varphi \in \mathscr{R}$, $\varphi \preceq \mathscr{R}^\vee$. Therefore, $\neg\mathscr{C}(\mathscr{R}^\vee, \mathscr{D}) \Rightarrow \neg\mathscr{C}(\varphi, \mathscr{D})$. ∎

> **Property 1.2 — Anti-monotone constraint evaluation.** Let $\mathscr{C}$ be an anti-monotone constraint. If $\mathscr{C}(\mathscr{R}^\wedge, \mathscr{D})$ is not satisfied, then $\forall \varphi \in \mathscr{R}$, $\mathscr{C}(\varphi, \mathscr{D})$ is not satisfied.

Proof. $\forall \varphi \in \mathscr{R}$, $\mathscr{R}^\wedge \preceq \varphi$. Therefore, $\neg\mathscr{C}(\mathscr{R}^\wedge, \mathscr{D}) \Rightarrow \neg\mathscr{C}(\varphi, \mathscr{D})$. ∎

Constraints that are piecewise monotone and anti-monotone can also be pushed, as it is explained below.

> **Property 1.3 — Piecewise monotone and anti-monotone constraint evaluation.** Let $\mathscr{C}$ be a piecewise monotone and anti-monotone constraint whose expression involves p times the pattern φ: $\mathscr{C}(\varphi, \mathscr{D}) = f(\varphi_1, \cdots \varphi_p, \mathscr{D})$. $\forall i \in \{1, \ldots, p\}$, the partial function $f_{i,\varphi}(x) = (\varphi_1, \cdots \varphi_{i-1}, x, \varphi_{i+1}, \cdots, \varphi_p, \mathscr{D})$ is either monotone or anti-monotone. Without lost of generality, let us consider that $\forall i \in \{1, \ldots, k\}$, $f_{i,\varphi}$ is monotone and $\forall i \in \{k+1, \ldots, p\}$, $f_{i,\varphi}$ is anti-monotone. Therefore, if $f(\underbrace{\mathscr{R}^\vee, \ldots, \mathscr{R}^\vee}_{k \text{ terms}}, \underbrace{\mathscr{R}^\wedge, \ldots, \mathscr{R}^\wedge}_{p-k \text{ terms}}, \mathscr{D})$ is not satisfied, then $\forall \varphi \in \mathscr{R}$, $\mathscr{C}(\varphi, \mathscr{D})$ is not satisfied.

Proof. Suppose there exists $\varphi \in \mathscr{R}$ such that $\mathscr{C}(\varphi, \mathscr{D})$ is satisfied. Then, $\mathscr{R}^\wedge \preceq \varphi \preceq \mathscr{R}^\vee$ and $f(\varphi_1, \cdots \varphi_p, \mathscr{D})$ is satisfied. $\forall i \in \{1, \ldots, k\}$, $f_{i,\varphi}(\mathscr{R}^\vee)$ is satisfied and $\forall i \in \{k+1, \ldots, p\}$, $f_{i,\varphi}(\mathscr{R}^\wedge)$ is satisfied. Therefore, $f(\underbrace{\mathscr{R}^\vee, \ldots, \mathscr{R}^\vee}_{k \text{ terms}}, \underbrace{\mathscr{R}^\wedge, \ldots, \mathscr{R}^\wedge}_{p-k \text{ terms}}, \mathscr{D})$ is satisfied. ∎

Constraint propagation mechanisms

The propagation mechanisms are as follows:

Monotone constraints: If there exists an element $v \in \mathscr{R}^\vee \setminus \mathscr{R}^\wedge$ such that $\neg\mathscr{C}((\mathscr{R}^\vee \setminus \{v\}), \mathscr{D})$ then v can be moved into $\mathscr{R}^\wedge$.

■ **Example 1.11** Let v be a positive real-valued attribute associated to the elements of φ and let $\mathscr{C}(\varphi, v) \equiv \operatorname{sum}(v(\varphi)) \geq \sigma$, which states that the sum of φ on v should be greater than σ, with $\operatorname{sum}(v(\varphi)) = \sum_{e \in \varphi} v(e)$. This constraint is monotone. Suppose now that $\mathscr{R}^\wedge = \{e_1, e_3, e_4\}$ and $\mathscr{R}^\vee = \{e_1, e_2, e_3, e_4, e_5\}$ with $v(e_1) = 2$, $v(e_2) = 8$, $v(e_3) = 4$, $v(e_4) = 3$ and $v(e_5) = 13$. If $\sigma = 20$, then $\mathscr{C}((\{e_1, e_2, e_3, e_4, e_5\} \setminus \{e_2\}), \mathscr{D}) \equiv 22 \geq 20$ and consequently e_2 is not propagated. As $\mathscr{C}((\{e_1, e_2, e_3, e_4, e_5\} \setminus \{e_5\}), \mathscr{D}) \equiv 17 \geq 20$, e_5 can be moved into $\mathscr{R}^\wedge$. ■

Anti-monotone constraints: If there exists $v \in \mathscr{R}^\vee \setminus \mathscr{R}^\wedge$ such that $\neg \mathscr{C}((\mathscr{R}^\wedge \cup \{v\}), \mathscr{D})$ then v can be removed from $\mathscr{R}^\vee$.

■ **Example 1.12** Let v be a positive real-valued attribute associated to the elements of φ and let $\mathscr{C}(\varphi, v) \equiv \mathrm{sum}(v(\varphi)) \leq \sigma$, which states that the sum of φ on v should be smaller than σ, with $\mathrm{sum}(v(\varphi)) = \sum_{e \in \varphi} v(e)$. This constraint is anti-monotone. Suppose now that $\mathscr{R}^\wedge = \{e_1, e_3, e_4\}$ and $\mathscr{R}^\vee = \{e_1, e_2, e_3, e_4, e_5\}$ with $v(e_1) = 2$, $v(e_2) = 8$, $v(e_3) = 4$, $v(e_4) = 3$ and $v(e_5) = 13$. If $\sigma = 20$, then $\mathscr{C}((\{e_1, e_3, e_4\} \cup \{e_2\}), \mathscr{D}) \equiv 17 \leq 20$ and consequently e_2 is not propagated. As $\mathscr{C}((\{e_1, e_3, e_4\} \cap \{e_5\}), \mathscr{D}) \equiv 22 \leq 20$, e_5 can be removed from $\mathscr{R}^\vee$.   ■

Piecewise monotone and anti-monotone constraints: If there exists an element $v \in \mathscr{R}^\vee \setminus \mathscr{R}^\wedge$ such that $\neg f(\underbrace{\mathscr{R}^\vee \setminus \{v\}, \ldots, \mathscr{R}^\vee \setminus \{v\}}_{k\ terms}, \underbrace{\mathscr{R}^\wedge, \ldots, \mathscr{R}^\wedge}_{p-k\ terms}, \mathscr{D})$ then v can be moved into $\mathscr{R}^\wedge$. If there exists an element $v \in \mathscr{R}^\vee \setminus \mathscr{R}^\wedge$ such that $\neg f(\underbrace{\mathscr{R}^\vee, \ldots, \mathscr{R}^\vee}_{k\ terms}, \underbrace{\mathscr{R}^\wedge \cup \{v\}, \ldots, \mathscr{R}^\wedge \cup \{v\}}_{p-k\ terms}, \mathscr{D})$ then v can be removed from $\mathscr{R}^\vee$.

■ **Example 1.13** Let v be a positive real-valued attribute associated to the elements of φ, $\mathscr{C}(\varphi, \mathscr{D}) = \mathrm{avg}(v(\varphi)) \geq \sigma$, and $\mathrm{avg}(v(\varphi)) = \frac{\sum_{e \in \varphi} v(e)}{|\varphi|}$. $\mathscr{C}(\varphi, \mathscr{D})$ is piecewise monotone and anti-monotone with $\mathscr{C}(\varphi, \mathscr{D}) = f(\varphi_1, \varphi_2, \mathscr{D})$ and

$$f(\varphi_1, \varphi_2, \mathscr{D}) = \frac{\sum_{e \in \varphi_1} v(e)}{|\varphi_2|}$$

with $f_{1,\varphi}$ monotone and $f_{2,\varphi}$ anti-monotone. Suppose now that $\mathscr{R}^\wedge = \{e_1, e_3, e_4\}$ and $\mathscr{R}^\vee = \{e_1, e_2, e_3, e_4, e_5\}$ with $v(e_1) = 2$, $v(e_2) = 8$, $v(e_3) = 4$, $v(e_4) = 3$ and $v(e_5) = 20$. If $\sigma = 8$, then

- $f(\mathscr{R}^\vee \setminus \{e_2\}, \mathscr{R}^\wedge, \mathscr{D}) \equiv \frac{\sum_{e \in \{e_1, e_3, e_4, e_5\}} v(e)}{|\{e_1, e_3, e_4\}|} \geq \sigma \equiv \frac{29}{3} \geq 8 \equiv 9.66 \geq 8$ and e_2 is not propagated;
- $f(\mathscr{R}^\vee \setminus \{e_5\}, \mathscr{R}^\wedge, \mathscr{D}) \equiv \frac{\sum_{e \in \{e_1, e_2, e_3, e_4\}} v(e)}{|\{e_1, e_3, e_4\}|} \geq \sigma \equiv \frac{17}{3} \geq 8 \equiv 5.66 \geq 8$ and e_5 is moved into $\mathscr{R}^\wedge$;
- Now, we have $\mathscr{R}^\wedge = \{e_1, e_3, e_4, e_5\}$ and $f(\mathscr{R}^\vee, \mathscr{R}^\wedge \cup \{e_2\}, \mathscr{D}) \equiv \frac{\sum_{e \in \{e_1, e_2, e_3, e_4, e_5\}} v(e)}{|\{e_1, e_2, e_3, e_4, e_5\}|} \geq \sigma \equiv \frac{37}{5} \geq 8 \equiv 7.4 \geq 8$ and e_2 is removed from $\mathscr{R}^\vee$.   ■

1.4.3 Pseudo-code

Algorithm 2 presents the steps of the generic algorithm. Lines 1 and 2 initialize $\mathscr{R}$ join value to the lattice top and the meet value to the lattice bottom. Line 3 is the first call to `Generic_CBPM_enumeration` function which enumerates once and only once each pattern. The first line of the function evaluates the constraints on the meet and join elements of $\mathscr{R}$. If so, line 2 tests if the search space contains a single pattern that is output line 3. Line 5 reduces the search space by removing elements whose enumeration will emptied the search or moving elements in $\mathscr{R}^\wedge$ as they are mandatory for the constraint satisfaction. Then, line 6, a new element, that belongs to the join but not to the meet, is enumerated according to the binary partition algorithm. This element is first added to the search space meet in the recursive call line 7, and then it is removed from the search space join in the second recursive call line 8.

1.5 Case studies

These generic algorithmic principles have been used in different contexts. In the following I present their use in two distinct pattern domain mining, for which pattern languages and specific

---

Algorithm 2 `Generic_CBPM`

Require: The data $\mathscr{D} \subseteq \mathscr{D}^1 \times \cdots \times \mathscr{D}^d$, a language of set patterns $\mathscr{L} \subseteq 2^{\mathscr{D}^1} \times \cdots \times 2^{\mathscr{D}^d}$, $\mathscr{C}$ a conjunction of monotone, anti-monotone and piecewise monotone and anti-monotone constraints.

Ensure: $\mathscr{T} = \mathrm{Th}(\mathscr{L}, \mathscr{D}, \mathscr{C})$

  1: $\mathscr{R}^\vee \leftarrow \mathscr{D}^1 \times \cdots \times \mathscr{D}^d$

  2: $\mathscr{R}^\wedge \leftarrow \emptyset \times \cdots \times \emptyset$

  3: $\mathscr{T} \leftarrow$ `Generic_CBPM_Enumeration`$(\mathscr{R}^\vee, \mathscr{R}^\wedge)$

Function `Generic_CBPM_enumeration`$(\mathscr{R}^\vee, \mathscr{R}^\wedge)$

  1: **if** `Check_constraints`$(\mathscr{R}^\wedge, \mathscr{R}^\vee)$ **then**

  2:     **if** $\mathscr{R}^\wedge = \mathscr{R}^\vee$ **then**

  3:        $\mathscr{T} \leftarrow \mathscr{T} \cup \mathscr{R}^\wedge$

  4:     **else**

  5:        $(\mathscr{R}^\wedge, \mathscr{R}^\vee) \leftarrow$ `Constraint_Propagation`$(\mathscr{R}^\wedge, \mathscr{R}^\vee)$

  6:        **for all** $e \in \mathscr{R}^\vee \setminus \mathscr{R}^\wedge$ **do**

  7:           $\mathscr{T} \leftarrow \mathscr{T} \cup$ `Generic_CBPM_Enumeration`$(\mathscr{R}^\wedge \cup \{e\}, \mathscr{R}^\vee)$

  8:           $\mathscr{T} \leftarrow \mathscr{T} \cup$ `Generic_CBPM_Enumeration`$(\mathscr{R}^\wedge, \mathscr{R}^\vee \setminus \{e\})$

  9:        **end for**

10:     **end if**

11: **end if**

12: **return** $\mathscr{T}$

---

constraints have been implemented: N-sets mining in n-ary Relations (Cerf, Besson, Robardet, and Boulicaut 2008; Cerf, Besson, Robardet, and Boulicaut 2009), a work done during the PhD of Loïc Cerf, and parallel episode mining in sequence database (Cule, Goethals, and Robardet 2009), realized with Boris Cule during his PhD. Chapter 3 presents their use for dynamic attribute graph mining.

1.5.1   Closed pattern mining in n-ary Relations

In (Cerf, Besson, Robardet, and Boulicaut 2008; Cerf, Besson, Robardet, and Boulicaut 2009) we consider the problem of extracting closed patterns in *n*-ary relations. Associations relating multiple types of instances are often represented by multi-way arrays (also known as tensors). As an example, we can consider sales data that gather information about the products, the regions, and the weeks of customer transactions. These data form a three-ary relation where the first dimension represents the products, the second dimension represents the regions, and the third dimension represents the weeks. It generalizes to *n*-ary relations the concept of itemsets (Agrawal and Srikant 1994), that has been extensively studied during the last decade. Closed patterns in *n*-ary relations reveal associations between groups of instances and thereby yield insights into the higher-level association structure in the data. In our above example, such patterns would detect maximal groups of products that were popular in certain regions during a number of weeks.

Dataset and language of patterns

In this context, the dataset is a *n*-ary data array, or tensor. Let $n > 0$ be the number of dimensions in the given array and $A^1, \cdots, A^n$ be n discrete-valued attributes whose domains are respectively $D^1, \cdots, D^n$. The *n*-ary relation $\mathscr{D}$ is such as

$$\mathscr{D} \subseteq D^1 \times \cdots \times D^n$$

From this dataset, we are interested in extracting n-sets, defined by specifying, for each dimension, a non-empty subset. The language of patterns is thus

$$\mathscr{L} = \{\varphi = (\varphi^1, \cdots, \varphi^n) \mid \varphi^i \subseteq D^i, |\varphi^i| \geq 1, i = 1, \cdots, n\}$$

Definition of the constraints

To identify closed n-sets as an extension of formal concepts to n-ary relations, we introduce two constraints $\mathscr{C}_{\text{CONNECTED}}$ and $\mathscr{C}_{\text{CLOSED}}$. $\mathscr{C}_{\text{CONNECTED}}(\varphi, \mathscr{D})$ guarantees that all elements of each set φ^i are in relation with all the other elements of the other sets in $\mathscr{D}$, and $\mathscr{C}_{\text{CLOSED}}(\varphi, \mathscr{D})$ certifies that adding an element to any dimension of φ violates the $\mathscr{C}_{\text{CONNECTED}}$ constraint. Formally,

$$\mathscr{C}_{\text{CONNECTED}}(\varphi, \mathscr{D}) \equiv \forall X = (x^1, \cdots, x^n), x^i \in \varphi^i, X \in \mathscr{D}$$
$$\mathscr{C}_{\text{CLOSED}}(\varphi, \mathscr{D}) \equiv \forall y^j \in D^j \setminus \varphi^j, j \in \{1, \cdots, n\}, \exists x^i \in \varphi^i, \forall i \neq j \text{ such that}$$
$$(x^1, \ldots, x^{j-1}, y^j, x^{j+1}, \ldots, x^n) \notin \mathscr{D}$$

To guide the search toward most interesting patterns, we consider two other constraints that coerce the patterns to contain a minimal number of elements ($\mathscr{C}_{\text{SIZE}}$) or to have a minimal volume ($\mathscr{C}_{\text{VOLUME}}$):

$$\mathscr{C}_{\text{SIZE}}(\varphi, \mathscr{D}) \equiv \wedge_{i=1}^n |\varphi^i| \geq \alpha_i$$
$$\mathscr{C}_{\text{VOLUME}}(\varphi, \mathscr{D}) \equiv \Pi_{i=1}^n |\varphi^i| \geq \alpha$$

However, for a given closed n-set, there may exist outside elements that are in relation with "almost" all inside elements. Such patterns are not "isolated". To avoid the extraction of such n-sets, we propose a new constraint: $\mathscr{C}_{\text{ISOLATED}}$. We first consider the projection of a n-set φ on an element $y^j \in D^j$:

$$\text{Projection}(\varphi, y^j) = (\varphi^1 \times \cdots \times \varphi^{j-1} \times \{y^j\} \times \varphi^{j+1} \times \cdots \varphi^n)$$

A projection is a kind of hyper plane of φ whose dimension j is fixed. By considering the proportion of elements of this projection that belongs to $\mathscr{D}$ to the total number of elements of the projection, we obtain the density of y^j on φ. The less this density, the more isolated φ:

$$\mathscr{C}_{\text{ISOLATED}}(\varphi, \mathscr{D}) \equiv \max_j \max_{y^j \in D^j \setminus \varphi^j} \frac{|\text{Projection}(\varphi, y^j) \cap \mathscr{D}|}{|\text{Projection}(\varphi, y^j)|} \leq \alpha$$

$\alpha \in [0, 1]$ is a user-defined parameter. When $\alpha = 0$, any element of D^j outside the n-set are not in relation with elements from the $(D^i)_{j \neq i}$ contained in the n-set.

Illustrative example

Figure 1.2 provides a ternary relation $\mathscr{D}_E$. $\langle (A, B), (1, 2), (\alpha, \gamma) \rangle$ and $\langle (C), (4), (\alpha, \beta, \gamma) \rangle$ are examples of 3-sets in $\mathscr{D}_E$. The 3-set $\langle (A, B), (1, 2, 3), (\alpha, \gamma) \rangle$ violates $\mathscr{C}_{\text{CONNECTED}}$ because $(A, 3, \alpha) \notin \mathscr{D}_E$ or $(B, 3, \gamma) \notin \mathscr{D}_E$. $\langle (C), (3, 4), (\beta) \rangle$ satisfies $\mathscr{C}_{\text{CONNECTED}}$ but not $\mathscr{C}_{\text{CLOSED}}$ because $(C, 1, \beta) \in \mathscr{D}_E$ or $(C, 3, \gamma) \in \mathscr{D}_E \wedge (C, 4, \gamma) \in \mathscr{D}_E$. Considering the 3-set $H = \langle (A, B, C), (1), (\alpha, \beta) \rangle$, $\mathscr{C}_{\text{ISOLATED}}(H, \mathscr{D})$ is true with $\alpha = 0.5$ but false with $\alpha = 0.4$ because of elements 2 and 4.

Constraint properties

Let $\mathscr{L}$ be partially ordered by the set inclusion, that is to say $\varphi_1 \preceq \varphi_2$ iff $\varphi_1^i \subseteq \varphi_2^i, \forall i = 1 \ldots n$, the constraints $\mathscr{C}_{\text{CONNECTED}}(\varphi, \mathscr{D}), \mathscr{C}_{\text{CLOSED}}(\varphi, \mathscr{D}), \mathscr{C}_{\text{SIZE}}(\varphi, \mathscr{D}), \mathscr{C}_{\text{VOLUME}}(\varphi, \mathscr{D})$ and $\mathscr{C}_{\text{ISOLATED}}(\varphi, \mathscr{D})$ satisfy the following properties.

	A	B	C	A	B	C	A	B	C
1	1	1	1	1	1	1	1	1	
2	1	1		1			1	1	
3		1					1	1	1
4			1	1			1	1	1
	α			β			γ		

Figure 1.2: Boolean representation of a relation $\mathscr{D}_E \subseteq \{A,B,C\} \times \{1,2,3,4\} \times \{\alpha,\beta,\gamma\}$

Property 1.4 $\mathscr{C}_{\text{CONNECTED}}(\varphi,\mathscr{D})$ is anti-monotone with respect to $\preceq$.

Proof. For all $\varphi_1,\varphi_2 \in \mathscr{L}$ such that $\varphi_1 \preceq \varphi_2$, if $\mathscr{C}_{\text{CONNECTED}}(\varphi_1,\mathscr{D})$ is not satisfied, then there exists $(x^1,\ldots,x^n)$, $x^i \in \varphi_1^i$ such that $(x^1,\ldots,x^n) \notin \mathscr{D}$. However, $\forall i$, $x^i \in \varphi_2^i$ and $\mathscr{C}_{\text{CONNECTED}}(\varphi_2,\mathscr{D})$ is not satisfied. ∎

Property 1.5 $\mathscr{C}_{\text{SIZE}}(\varphi,\mathscr{D})$ and $\mathscr{C}_{\text{VOLUME}}(\varphi,\mathscr{D})$ are monotone constraints.

Proof. For all $\varphi_1,\varphi_2 \in \mathscr{L}$ such that $\varphi_1 \preceq \varphi_2$, if $\mathscr{C}_{\text{SIZE}}(\varphi_1,\mathscr{D})$ (resp. $\mathscr{C}_{\text{VOLUME}}(\varphi_1,\mathscr{D})$) is satisfied, then $\mathscr{C}_{\text{SIZE}}(\varphi_2,\mathscr{D})$ (resp. $\mathscr{C}_{\text{VOLUME}}(\varphi_2,\mathscr{D})$) is also satisfied since it contains more elements than φ_1. ∎

Property 1.6 $\mathscr{C}_{\text{ISOLATED}}(\varphi,\mathscr{D})$ is piecewise monotone and anti-monotone with $\mathscr{C}_{\text{ISOLATED}}(\varphi,\mathscr{D}) = f(\varphi_1,\varphi_2,\varphi_3,\mathscr{D})$ and

$$f(\varphi_1,\varphi_2,\varphi_3,\mathscr{D}) = \max_j \max_{y^j \in D^j \setminus \varphi_1^j} \frac{|\text{Projection}(\varphi_2,y^j) \cap \mathscr{D}|}{|\text{Projection}(\varphi_3,y^j)|}$$

$f_{1,\varphi}$ and $f_{3,\varphi}$ are monotone, whereas $f_{2,\varphi}$ is anti-monotone.

Proof. For all $x,y \in \mathscr{L}$ such that $x \preceq y$,
1. $D^j \setminus x \supseteq D^j \setminus y$ and

$$f_{1,\varphi}(x) = \max_j \max_{y^j \in D^j \setminus x} \frac{|\text{Projection}(\varphi_2,y^j) \cap \mathscr{D}|}{|\text{Projection}(\varphi_3,y^j)|} \geq \max_j \max_{y^j \in D^j \setminus y} \frac{|\text{Projection}(\varphi_2,y^j) \cap \mathscr{D}|}{|\text{Projection}(\varphi_3,y^j)|}$$

Therefore, $f_{1,\varphi}(y) > \alpha$ implies $f_{1,\varphi}(x) > \alpha$ and $f_{1,\varphi}$ is monotone.
2. $|\text{Projection}(x,y^j)| \leq |\text{Projection}(y,y^j)|$ and

$$f_{2,\varphi}(x) = \max_j \max_{y^j \in D^j \setminus \varphi_1^j} \frac{|\text{Projection}(x,y^j) \cap \mathscr{D}|}{|\text{Projection}(\varphi_3,y^j)|} \leq \max_j \max_{y^j \in D^j \setminus \varphi_1^j} \frac{|\text{Projection}(y,y^j) \cap \mathscr{D}|}{|\text{Projection}(\varphi_3,y^j)|}$$

Therefore, $f_{2,\varphi}(x) > \alpha$ implies $f_{2,\varphi}(y) > \alpha$ and $f_{2,\varphi}$ is anti-monotone.
3. $1 \leq |\text{Projection}(x,y^j)| \leq |\text{Projection}(y,y^j)|$ and $\frac{1}{|\text{Projection}(y,y^j)|} \leq \frac{1}{|\text{Projection}(x,y^j)|} \leq 1$

$$f_{3,\varphi}(x) = \max_j \max_{y^j \in D^j \setminus \varphi_1^j} \frac{|\text{Projection}(\varphi_2,y^j) \cap \mathscr{D}|}{|\text{Projection}(y,y^j)|} \leq \max_j \max_{y^j \in D^j \setminus \varphi_1^j} \frac{|\text{Projection}(\varphi_2,y^j) \cap \mathscr{D}|}{|\text{Projection}(x,y^j)|}$$

Therefore, $f_{3,\varphi}(y) > \alpha$ implies $f_{3,\varphi}(x) > \alpha$ and $f_{3,\varphi}$ is monotone. ∎

Experiments

The instanciation of the generic algorithm to this pattern domain is called DATA-PEELER. To study the behavior of DATA-PEELER and to compare it to competitors, we have used the IBM QUEST data generator (Agrawal and Srikant 1995). Various data sets with predefined densities have been generated that are made of three attributes: The customers, the bought items, and the time periods (in months). We also have been working on logs from the `DistroWatch.com` website. This popular website gathers comprehensive information about GNU/Linux and BSD distributions. Every distribution being described on a separate page, a visitor loading such a page is considered "interested" in the distribution. Every IP address is analyzed to identify the country it comes from. Time steps allow to study the evolution of the interest granted to the different distributions along time. The whole data set gathers 36 months, 243 countries and 538 distributions. Two different data sets have been derived from it. In both cases, data have been normalized so that every country and every time period has the same importance. They have been transformed in 0/1 data in the following way: For each distribution, we kept the elements of $\mathscr{D}$ containing this distribution and such that its normalized interest exceeds a threshold equals to one quarter of the maximal normalized interest for this distribution in $\mathscr{D}$.

Comparisons with competitors: DATA-PEELER is compared to CUBEMINER (Ji, Tan, and Anthony 2006) and TRIAS (Jäschke, Hotho, Schmitz, Ganter, and Stumme 2006) on 3-ary synthetic data sets generated by QUEST, using the implementations provided by their authors. The data sets simulate 144 customers buying in average 6 items out of 72 (density of about 8.3%) per month. We make the number of months vary from 6 to 66 and constrain every closed 3-set to gather at least 2 customers, 2 items and 2 months. The results are represented in Figure 1.3 (left).
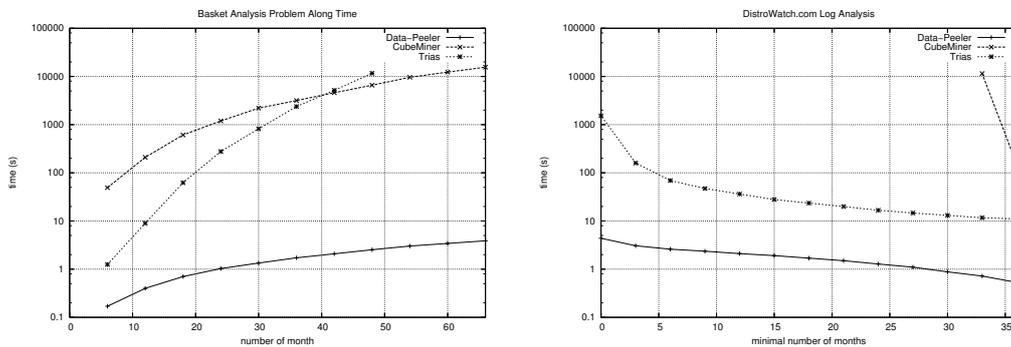
Figure 1.3: Comparison w.r.t. CUBEMINER and TRIAS on synthetic data (left) and real data (right).

DATA-PEELER outperforms its competitors by several orders of magnitude. The growing number of months (the smallest domain) consequently alters the performance of TRIAS, whereas it has less effect on CUBEMINER. For example, considering data along 48 months, to extract all the 5801 valid closed n-sets, CUBEMINER takes 1 hour and 50 minutes, TRIAS 3 hours and 14 minutes, whereas DATA-PEELER only needs 2.5 seconds. Unlike its competitors, even with 600 months, DATA-PEELER is still able to extract all valid closed n-sets in a reasonable time, i.e., 1 minute and 21 seconds for 431'892 closed n-sets.

The data set derived from the logs of `DistroWatch.com` indicates, month after month, whether visitors from a country seem interested in a distribution. All data (36 months, 243 countries and 538 distributions) have been kept. Compared to the synthetic data sets considered above, this one is relatively large even if it has one small attribute domain. It is also much sparser since its density is 0.55%. We have constrained every closed 3-set to gather at least 2 countries and 2 distributions. The minimal number of months a closed 3-set must contain has been varying

from 0 to 36. The results are represented in Figure 1.3 (right). DATA-PEELER outperforms its competitors by several orders of magnitude. Thanks to the small number of months in the data set, TRIAS succeeds in extracting the closed 3-sets even without any constraint on the time attribute. CUBEMINER suffers a lot from the global size of the data set. It is unable to perform the extraction under a size constraint of 33/36 months. For example, with a minimal size constraint of 6 months, DATA-PEELER needs 2.6 seconds and TRIAS 69.3 seconds to extract all the 87 patterns. Without any minimal size constraint on the number of months, 10 658 closed n-sets are computed in less than 4.4 seconds by DATA-PEELER and in 1531 seconds by TRIAS. In both cases, CUBEMINER can not perform the task.

A real-life application: We considered an interesting 4-ary relation from `DistroWatch.com` logs. It takes in consideration visitors who have loaded at least two different distribution pages the same day. We assumed that this is a sign of a common interest in the visited distributions. The less relevant countries and distributions have been removed. The days have been aggregated in semesters (the release period of many distributions). In the end, we obtained a 4-ary relation $\mathscr{D}_{DW}(D_1, D_2, S, C)$ indicating that people from country C (among 39) show a common interest in distributions D_1 and D_2 (among 323) during the semester S (among 6). This relation covers 1.7% of the possible associations between attributes. We extracted all the maximal sets of distributions which are simultaneously interesting for people from a maximal set countries during a maximal set of semesters. To obtain them, we mined the closed 4-sets $\langle X_1, X_2, X_3, X_4 \rangle \in 2^{D_1} \times 2^{D_2} \times 2^S \times 2^C$ of $\mathscr{D}_{DW}$ such that $X_1 = X_2$. This additional constraint is anti-monotone. DATA-PEELER extracted every closed 4-set from the data set in 229 seconds. Since it is impossible to inspect all the 602 290 closed 4-set, minimal sizes are enforced on every set: We constrained every closed 4-set to gather at least 2 semesters, 2 countries and 3 distributions. After 20 seconds, DATA-PEELER returned 17 196 closed 4-sets. Again, it prevents from a systematic interpretation of each closed 4-set. Therefore, we enforced a volume constraint to keep only the largest patterns. It reduced the computation to 14 seconds and the number of extracted closed 4-sets to 352. Given such a collection of 352 patterns, it was possible to manually inspect them and assess their relevancy.

- 94 closed 4-sets having on the distribution domains a subset of {Fedora, FreeBSD, Debian, Ubuntu, Gentoo, MEPIS, Slackware, Yellow Dog, Mandriva, openSUSE}. Considering all of them, every semester is mentioned. All these distributions are mainstream general-purpose distributions. These closed 4-sets gather many countries all over the world. However Great Britain shows off by being present in all these n-sets but one.
- 64 closed 4-sets having on the distribution domains a subset of {Astaro, ClarkConnect, IPCop, m0n0wall, Devil, SmoothWall, CensorNet}. Every semester is involved. These seven distributions are meant to serve a common interest: they are all specifically designed to act as a firewall. These closed 4-sets gather countries from every continent. Australia (closely followed by Belgium) is the most present country.
- 80 4-sets having on the distribution domains a subset of {dyne:bolic, ArtistX, AGNULA, MoviX, GeeXboX}. Every semester is involved. These five distributions are meant to serve a common interest: they are all specifically designed to manipulate movies and music. These 4-sets mainly contain occidental countries but India is very present too. Switzerland belongs to all these 4-sets. GNU/Linux is obviously a popular choice among Swiss artists.
- 83 closed 4-sets having on the distribution domains a subset of {dyne:bolic, ArtistX, AGNULA, MoviX, GeeXboX} ∪ {Ubuntu, Damn Small, KNOPPIX, MEPIS, PCLinuxOS, Xandros}. It could be seen as a "collision" between two separate interests. Nevertheless, the distributions from the second set being primarily designed for desktop use, they are also suited to play movies and music. Furthermore, every distribution from these two sets uses the APT package management system (if any).

The few remaining closed 4-sets are not out of interest. Among the distributions which appear once in the returned closed 4-sets, Gentoox and GentooTH form with Gentoo a closed 4-set running along the last four semesters (GentooTH did not exist before) in 11 countries. Their common point is obvious from their names: they are all based on Gentoo. In the same way, Knopperdisk and Feather are mentioned in one single closed 4-set where they are associated with Damn Small along the last five semesters (Knopperdisk did not exist before). These distributions have a strong common point: all of them are KNOPPIX light derivatives aimed at being installed on a USB pen drive.

Instead of searching for large closed patterns, we now focus on extracting closed 4-sets which are isolated in the country domain using $\mathscr{C}_{\text{ISOLATED}}$ constraint. We first set $\delta = 0.25$ on the country attribute. Keeping the size constraints from the previous section, DATA-PEELER returns one single closed 4-set. It gathers, during two semesters, the distributions B2D, Linpus and PUD for Taiwan and Hong Kong. These three distributions are Taiwanese and insist on the direct support of traditional Chinese. Traditional Chinese characters are almost exclusively used in Taiwan, Hong Kong and Macao (which was not kept among the 39 countries in the data set). Indeed, the impact of this strong particularity is well revealed by enforcing the $\mathscr{C}_{\text{ISOLATED}}$ constraint. When increasing δ to 0.3, DATA-PEELER returns two additional closed 4-sets. Both of them refer to Russia and Ukraine during 2006 and gather ALT, ASP and one mainstream distribution (either Ubuntu or Mandriva). Both ALT and ASP are Russian distributions. Again, the particularity of the Russian alphabet as the default character encoding is caught by the $\mathscr{C}_{\text{ISOLATED}}$ constraint.

1.5.2 Parallel episode mining in a sequence

In (Cule, Goethals, and Robardet 2009) we consider the problem of extracting parallel episodes in a single sequence. Discovering interesting episodes (Mannila, Toivonen, and Inkeri Verkamo 1997) is a popular area in temporal or sequential data mining, examples of which are text or protein sequences mining. In such data, the order in which the events appear is being analysed and the user's goal is to identify the regularities that may appear in the dataset, consisting of one or more sequences. The usual approach to episode discovery is to look for episodes consisting of events that frequently appear close to each other. Most of the current state-of-the-art methods first use a window of fixed length to find sufficiently cohesive episodes and then retrieve those that occur in more windows than a given minimum threshold (Mannila, Toivonen, and Inkeri Verkamo 1997). The use of a window of fixed length is a major limitation of such approaches as no episodes longer than this window can ever be discovered. A different method that increases the window length proportionally to the size of the candidate set has been proposed in order to overcome this limitation (Garriga 2003). Still, in this proposal, the window length remains fixed for a particular candidate when counting its frequency in the sequence. Hence, when the episode occurs in the sequence, but in a time frame larger than the window size, then such occurrences are disregarded. The high frequency of a set of events appearing close together gives no guarantee that a subset of that set will not sometimes appear far away from the rest of the set.

This observation motivates us to propose a new constraint to select interesting sets of events in a dataset consisting of a single sequence of events. We focus on parallel episodes which are unordered sets of events, and these can also be considered as sets of items, better known as itemsets. We define the interestingness of a parallel episode based on a combination of how often the events in the candidate set appear in the sequence and how close to each other they appear on average. Hence, this approach does not use a fixed window length but instead also takes the occurrences of the events far from the rest of the set into account. It is precisely such occurrences that might sufficiently lower the cohesion of an episode to render it uninteresting.

Dataset and language of patterns

In this context, the dataset $\mathscr{D}$ is a single sequence of events from $\mathscr{E}$, the set of all possible events. All consecutive events of the sequence are considered to be equidistant and thus the time steps of the sequence are considered to be the elements of the interval $[1,k]$ of natural numbers. As two events do not occur at the same time, the sequence is denoted by $\mathscr{D} = \langle e_1, \cdots, e_k \rangle$, where $e_i \in \mathscr{E}$, $i \in [1,k]$ and k is the number of events in $\mathscr{D}$. The mapping $TS(e, \mathscr{D})$ associates to each event e of $\mathscr{E}$ the set of the indices at which e appears:

$$TS(e, \mathscr{D}) = \{t \mid e_t \in \mathscr{D} \text{ and } e_t = e\}$$

As previously mentioned, we want to identify events that appear next to each other, but the order in which they occur does not matter. Therefore, the language of patterns we considered is the one of parallel episodes, that is the language of sets of events: $\mathscr{L} = \{\varphi \mid \varphi \subseteq \mathscr{E}\}$.

Definition of the constraints

To identify parallel episodes whose events appear frequently in the sequence and consist of events that, on average, appear close to each other, we introduce an interestingness measure that takes into account the number of occurrences of the events in $\mathscr{D}$ and the average length of the shortest time intervals in which all their events appear. The interestingness of a parallel episode depends therefore on two factors: its support and its cohesion. The support measures how often an event of the episode appears in $\mathscr{D}$, while cohesion measures how close together the events making up the episode appear on average. For a given parallel episode φ, we denote the set of all occurrences of its events as

$$\text{SUPPORT}(\varphi, \mathscr{D}) = \{\bigcup_{e \in \varphi} TS(e, \mathscr{D})\}$$

The coverage of φ can be defined as:

$$\text{COVERAGE}(\varphi, \mathscr{D}) = \frac{|\text{SUPPORT}(\varphi)|}{|\mathscr{D}|}$$

In order to calculate the cohesion of φ in $\mathscr{D}$, we must first evaluate the length of the shortest interval containing all the events of φ for a given time step t in $\mathscr{D}$:

$$Length(\varphi, t) = \min\{t_2 - t_1 + 1 \mid t_1 \leq t \leq t_2 \text{ and } \forall e \in \varphi, \exists t' \in TS(e, \mathscr{D}), t_1 \leq t' \leq t_2\}$$

We can now compute the average length of such shortest intervals for each time step in $\text{SUPPORT}(\varphi, \mathscr{D})$:

$$\overline{Length}(\varphi, \mathscr{D}) = \frac{\sum_{t \in \text{SUPPORT}(\varphi, \mathscr{D})} Length(\varphi, t)}{|\text{SUPPORT}(\varphi, \mathscr{D})|}$$

It is clear that $\overline{Length}(\varphi, \mathscr{D})$ is greater than or equal to the number of events in φ. Furthermore, for a fully cohesive parallel episode, where no other events ever occur between the events making the episode, $\overline{Length}(\varphi, \mathscr{D}) = |\varphi|$. To normalize, we want the cohesion of a fully cohesive episode to be equal to 1, and the cohesion of other episodes to be lower. Therefore, we define cohesion of φ as

$$\text{COHESION}(\varphi, \mathscr{D}) = \frac{|\varphi|}{\overline{Length}(\varphi, \mathscr{D})}$$

We can now define the interestingness of an episode φ as

$$
\begin{aligned}
\text{INTEREST}(\varphi, \mathscr{D}) &= \text{COHESION}(\varphi, \mathscr{D}) \times \text{COVERAGE}(\varphi, \mathscr{D}) \\
&= \frac{|\text{SUPPORT}(\varphi, \mathscr{D})| \times |\text{SUPPORT}(\varphi, \mathscr{D})| \times |\varphi|}{\sum_{t \in \text{SUPPORT}(\varphi, \mathscr{D})} Length(\varphi, t) \times |\mathscr{D}|}
\end{aligned}
$$

Note that both SUPPORT$(\varphi, \mathscr{D})$ and COHESION$(\varphi, \mathscr{D})$ are between 0 and 1, and the same is therefore true for INTEREST$(\varphi, \mathscr{D})$. This makes it possible to apply the same interestingness thresholds to any kind of dataset. Given three user defined thresholds, *min_support*, *min_cohesion* and *min_interest*, a parallel episode φ must satisfied

$$
\begin{aligned}
\mathscr{C}_{\text{COVERAGE}}(\varphi, \mathscr{D}) &\equiv \text{COVERAGE}(\varphi, \mathscr{D}) \geq \textit{min\_support} \\
\mathscr{C}_{\text{COHESION}}(\varphi, \mathscr{D}) &\equiv \text{COHESION}(\varphi, \mathscr{D}) \geq \textit{min\_cohesion} \\
\mathscr{C}_{\text{INTEREST}}(\varphi, \mathscr{D}) &\equiv \text{INTEREST}(\varphi, \mathscr{D}) \geq \textit{min\_interest}
\end{aligned}
$$

Illustrative example

In the example of Figure 1.4, *cd* and *cde* are both fully cohesive, but *cde* clearly covers more of the sequence than *cd*, and will therefore be considered more interesting. According to our definition, COVERAGE$(cd, \mathscr{D}) = \frac{4}{12}$ and COVERAGE$(cde, \mathscr{D}) = \frac{6}{12}$. As *cd* and *cde* are both fully cohesive, COHESION$(cd, \mathscr{D}) = 1$ and COHESION$(cde, \mathscr{D}) = 1$. Therefore, INTEREST$(cd, \mathscr{D}) = \frac{4}{12}$ and INTEREST$(cde, \mathscr{D}) = \frac{6}{12}$. In general, an episode will always have a lower coverage than any of its supersets, and, provided that they are equally cohesive, will always have a lower interestingness than the superset.

Time Steps	1	2	3	4	5	6	7	8	9	10	11	12
Sequence	a	b	c	d	e	f	g	c	d	e	h	i
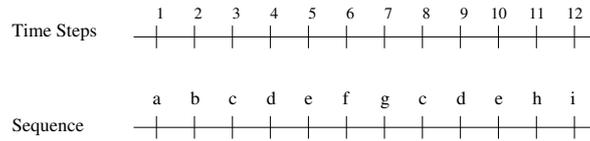
Figure 1.4: An illustrative example.

Let us now consider the results obtained by Winepi algorithm (Mannila, Toivonen, and Inkeri Verkamo 1997; Mannila, Toivonen, and Verkamo 1995), that finds all episodes that occur in a sufficient number of windows of fixed length. Applying WINEPI (with a window length greater than 1) to this sequence will result in *cd* having a larger frequency than *cde*. For example, using a window of length 3 gives $freq(cd) = \frac{4}{14}$ and $freq(cde) = \frac{2}{14}$. This is clearly unintuitive. Garriga (Garriga 2003) proposes to solve this problem by increasing the window length proportionally to the episode length, using a parameter *tus* equal to the maximal gap allowed between two events in an episode. Therefore, using *tus* = 2 (equivalent to using a window of length 3 for an episode made of two event types) would result in a window of length 5 for episodes made of three event types. Frequencies of *cd* and *cde* are in this context $fr(cd) = \frac{4}{14}$ and $fr(cde) = \frac{8}{16}$.

Time Steps	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Sequence	c	g	h	e	f	a	b	a	b	c	i	d	j	c	k	d	l	c	m	d
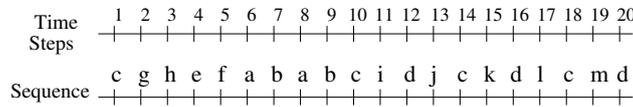
Figure 1.5: A second example

On the other hand, we should note that the cohesion of an episode alone is also not suitable for estimating its interestingness. In the example of Figure 1.5, *ef* is just as cohesive as *ab*, but its frequency is too small to be considered more interesting than *cd*. This is why our interestingness measure consists of combining both coverage and cohesion.

If we apply the WINEPI method on the example of Figure 1.5 with a window of length 3, we obtain the following frequencies: $fr(cd) = \frac{5}{22}$, $fr(ab) = \frac{4}{22}$ and $fr(ef) = \frac{2}{22}$. The ranking will be the same for all windows of length greater than 3. The same method with a window of

length 2 would give: $fr(cd) = 0$, $fr(ab) = \frac{3}{21}$ and $fr(ef) = \frac{1}{21}$. We can observe that no chosen window length gives the desired result. The method of Garriga (Garriga 2003) would give the same results depending on the used parameter. This is not a surprise, since both WINEPI and Garriga methods are trying to find how likely we are to encounter an episode, while it may be more interesting to find how likely we are to encounter an event from an episode, and, once we encountered it, how likely we are to encounter the other events of the episode nearby.

φ	$|\text{SUPPORT}(\varphi, \mathscr{D})|$	$\overline{Length}(\varphi, \mathscr{D})$	$\text{COHESION}(\varphi, \mathscr{D})$	$\text{COVERAGE}(\varphi, \mathscr{D})$	$\text{INTEREST}(\varphi, \mathscr{D})$
ab	4	2	1	0.2	0.2
cd	7	4.29	0.47	0.35	0.16
ef	2	2	1	0.1	0.1

Table 1.1: Computation of interestingness for example of Figure 1.5

The summary of the computation of the interestingness for the 3 itemsets previously considered is given in Table 1.1. Comparing these results with those obtained with WINEPI and Garriga, it can be observed that our method gives more intuitive results than WINEPI and Garriga.

Constraint properties

Given that $\mathscr{L}$ is partially ordered by the set inclusion, that is to say $\varphi_1 \preceq \varphi_2$ iff $\varphi_1 \subseteq \varphi_2$, the constraints $\mathscr{C}_{\text{COVERAGE}}(\varphi, \mathscr{D})$, $\mathscr{C}_{\text{COHESION}}(\varphi, \mathscr{D})$ and $\mathscr{C}_{\text{INTEREST}}(\varphi, \mathscr{D})$ satisfy the following properties.

> **Property 1.7** $\mathscr{C}_{\text{COVERAGE}}(\varphi, \mathscr{D})$ is monotone with respect to $\preceq$.

Proof. For all $\varphi_1, \varphi_2 \in \mathscr{L}$ such that $\varphi_1 \preceq \varphi_2$, we have $\text{SUPPORT}(\varphi_1, \mathscr{D}) \subseteq \text{SUPPORT}(\varphi_2, \mathscr{D})$ and $|\text{SUPPORT}(\varphi_1, \mathscr{D})| \leq |\text{SUPPORT}(\varphi_2, \mathscr{D})|$. Therefore, $\frac{|\text{SUPPORT}(\varphi_1)|}{|\mathscr{D}|} \geq min\_support$ implies $\frac{|\text{SUPPORT}(\varphi_2)|}{|\mathscr{D}|} \geq min\_support$. ∎

> **Property 1.8** $\mathscr{C}_{\text{COHESION}}(\varphi, \mathscr{D})$ is piecewise monotone and anti-monotone with $\mathscr{C}_{\text{COHESION}}(\varphi, \mathscr{D}) = f(\varphi_1, \varphi_2, \varphi_3, \varphi_4, \mathscr{D})$ and
>
> $$f(\varphi_1, \varphi_2, \varphi_3, \varphi_4, \mathscr{D}) = \frac{|\varphi_1||\text{SUPPORT}(\varphi_2, \mathscr{D})|}{\sum_{t \in \text{SUPPORT}(\varphi_3, \mathscr{D})} Length(\varphi_4, t)}$$
>
> $f_{1,\varphi}$ and $f_{2,\varphi}$ are monotone, whereas $f_{3,\varphi}$ and $f_{4,\varphi}$ are anti-monotone.

Proof. For all $x, y \in \mathscr{L}$ such that $x \preceq y$,
   1. $|x| \leq |y|$ and

$$f_{1,\varphi}(x) = \frac{|x||\text{SUPPORT}(\varphi_2, \mathscr{D})|}{\sum_{t \in \text{SUPPORT}(\varphi_3, \mathscr{D})} Length(\varphi_4, t)} \leq \frac{|y||\text{SUPPORT}(\varphi_2, \mathscr{D})|}{\sum_{t \in \text{SUPPORT}(\varphi_3, \mathscr{D})} Length(\varphi_4, t)} = f_{1,\varphi}(y)$$

Therefore, $f_{1,\varphi}(x) \geq min\_cohesion$ implies $f_{1,\varphi}(y) \geq min\_cohesion$ and $f_{1,\varphi}$ is monotone.
   2. $|\text{SUPPORT}(x, \mathscr{D})| \leq |\text{SUPPORT}(y, \mathscr{D})|$ and

$$f_{2,\varphi}(x) = \frac{|\varphi_1||\text{SUPPORT}(x, \mathscr{D})|}{\sum_{t \in \text{SUPPORT}(\varphi_3, \mathscr{D})} Length(\varphi_4, t)} \leq \frac{|\varphi_1||\text{SUPPORT}(y, \mathscr{D})|}{\sum_{t \in \text{SUPPORT}(\varphi_3, \mathscr{D})} Length(\varphi_4, t)} = f_{2,\varphi}(y)$$

Therefore, $f_{2,\varphi}(x) \geq min\_cohesion$ implies $f_{2,\varphi}(y) \geq min\_cohesion$ and $f_{2,\varphi}$ is monotone.

3. As $\text{SUPPORT}(x,\mathscr{D}) \subseteq \text{SUPPORT}(y,\mathscr{D})$ and $Length(\varphi,t) \geq 1$,
$$1 \leq \sum_{t \in \text{SUPPORT}(x,\mathscr{D})} Length(\varphi_4,t) \leq \sum_{t \in \text{SUPPORT}(y,\mathscr{D})} Length(\varphi_4,t)$$

$$\frac{|\varphi_1||\text{SUPPORT}(\varphi_2,\mathscr{D})|}{\sum_{t \in \text{SUPPORT}(y,\mathscr{D})} Length(\varphi_4,t)} \leq \frac{|\varphi_1||\text{SUPPORT}(\varphi_2,\mathscr{D})|}{\sum_{t \in \text{SUPPORT}(x,\mathscr{D})} Length(\varphi_4,t)}$$

and $f_{3,\varphi}$ is anti-monotone.

4. $\forall t \in [1,k]$, $Length(x,t) \leq Length(y,t)$. Therefore,
$$1 \leq \sum_{t \in \text{SUPPORT}(\varphi_3,\mathscr{D})} Length(x,t) \leq \sum_{t \in \text{SUPPORT}(\varphi_3,\mathscr{D})} Length(y,t)$$

$$\frac{|\varphi_1||\text{SUPPORT}(\varphi_2,\mathscr{D})|}{\sum_{t \in \text{SUPPORT}(\varphi_3,\mathscr{D})} Length(y,t)} \leq \frac{|\varphi_1||\text{SUPPORT}(\varphi_2,\mathscr{D})|}{\sum_{t \in \text{SUPPORT}(\varphi_3,\mathscr{D})} Length(x,t)}$$

and $f_{4,\varphi}$ is anti-monotone.

∎

Property 1.9 $\mathscr{C}_{\text{INTEREST}}(\varphi,\mathscr{D})$ is piecewise monotone and anti-monotone.

Proof. As a product of a monotone and a piecewise monotone and anti-monotone constraints, $\mathscr{C}_{\text{INTEREST}}(\varphi,\mathscr{D})$ is piecewise monotone and anti-monotone. ∎

Experiments

In this section, we present the results of our experiments performed on various synthetic and real-life datasets, and analyse their implications.

Synthetic datasets: Fully random datasets would not be suitable for our purposes because all itemsets of equal length would be likely to have similar interestingness values. A more suitable dataset would be one generated using a Markov chain model, which enables us to increase the likelihood of certain items appearing close together and thus forming interesting itemsets.

In our experiments, we use a memoryless model, also called a simple random walk, in which the next item depends only on its immediate predecessor. We can therefore easily describe such a model using a transition matrix. The transition matrix used to generate the dataset is given in Table 1.2. Our goal is to generate a sequence in which *abc* would be an interesting pattern hidden among occurrences of several other items denoted *x* in the table. When a transition leads to an occurrence of *x*, a random item is picked from a group of 25 items, not including *a*, *b* or *c*. Doing this ensures that none of these 25 items will form interesting itemsets.

	a	*b*	*c*	*x*
a	0.2	0.35	0.35	0.1
b	0.35	0.2	0.35	0.1
c	0.35	0.35	0.2	0.1
x	0.05	0.05	0.05	0.85

Table 1.2: Transition matrix defining a Markov model

Using this model, we generated a sequence of 2000 events. We then ran our algorithm varying the interestingness threshold *min_interest* from 0.9 down to 0.2, 0.05 at a time. In all experiments no thresholds were set for coverage and cohesion separately. The results of our experiments can be seen in Figures 1.6 (a) and (b). Figure 1.6 (a) shows that as long as there are few interesting itemsets to be found, our pruning method works very efficiently. The most interesting itemset *abc* is found using a relatively high interestingness threshold of 0.45.

No further interesting itemsets can be found without lowering the threshold below 0.3, which confirms the intuitiveness of our method. With the interestingness level of 0.25, itemsets *ab*, *ac* and *bc* are also discovered as interesting. Because of the uniform distribution of the remaining 25 items, the number of considered candidates grows significantly once the threshold is lowered further. We can observe in Figure 1.6 (b) that the execution time grows proportionally with the number of generated candidates indicating that the most interesting itemsets can be discovered in a reasonable amount of time. To determine the most appropriate interestingness threshold value for a given dataset, an inexperienced user can start with a high value and decrease it until the first results come through. Running the algorithm with a high interestingness threshold takes only few seconds.
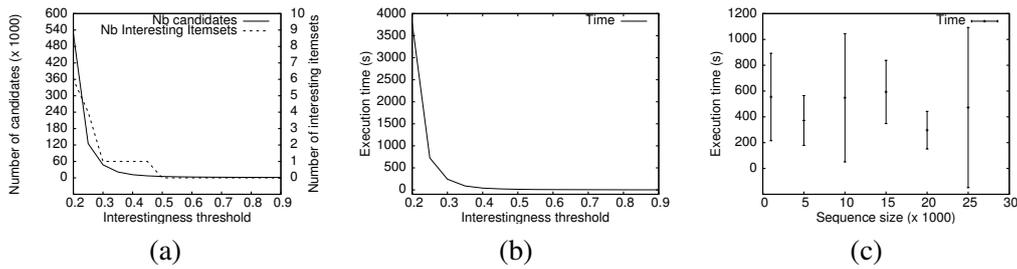
Figure 1.6: MARKOV CHAIN generated data: (a) Number of generated candidates and interesting episodes with varying *min_interest*; (b) Execution time in seconds with varying *min_interest*; (c) Execution time in seconds when *min_interest* = 0.3 and the sequence length is varying.

Analyzing the generated dataset, we observed that the coverage of *abc* was around 0.6. Its interestingness was around 0.46, which means its cohesion was around 0.77. In order to show that the high interestingness of *abc* was not only due to its coverage, we investigate what is obtained in a randomly ordered sequence in which items have the same frequency as in the Markov Chain dataset. We generated such a sequence of 2000 events made of 28 items. As can be seen in Figure 1.7 (a), we have to lower the interestingness threshold much further in order to get the first results. Episode *abc* is found to have interestingness of around 0.3 indicating a cohesion of 0.5. Figure 1.7 (b) shows that, once again, the execution time is fully proportional with number of generated candidates.
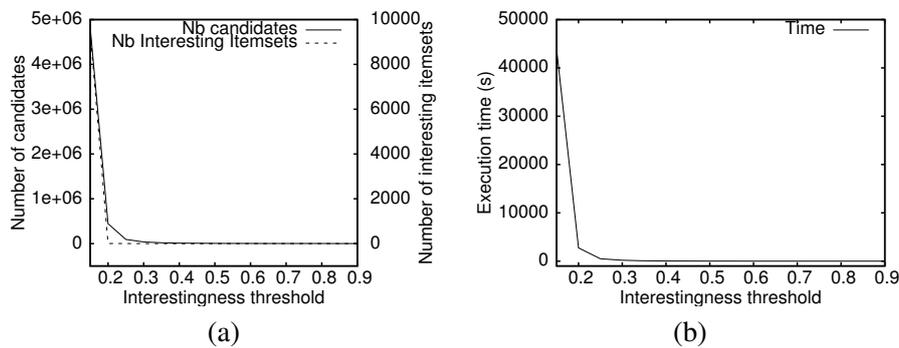
Figure 1.7: RANDOM dataset: (a) Number of generated candidates and interesting episodes with varying *min_interest*; (b) Execution time in seconds with varying *min_interest*.

To test how much the execution time depends on the size of the sequence, we generated several Markov chain datasets of different sizes, varying between 1000 and 25000. For each

size, we generated 10 datasets and applied our algorithm with *min_interest* = 0.3, because it is around this threshold that the first results usually appeared. Figure 1.6 (c) shows the obtained results. Each vertical line corresponds to one sequence length showing the mean execution time of the 10 algorithm runs and their standard deviation. It can be observed that the execution time depends much more on the structure of the dataset than on its size, even for such similarly structured datasets (all the datasets contained exactly the same number of different items that were distributed according to the same transition matrix).

Real-life datasets: The first real-life dataset we used is one obtained from the amino-acid sequence from human alcohol dehydrogenase (G. Wu 2000). This enzyme has been heavily researched and its genomic sequence is widely available and therefore reliable[1]. The sequence consists of 20 different amino-acids making a chain of 375. In Figure 1.8 (a), we can see that we get some results with an interestingness value below 0.4, but no itemset really sticks out, suggesting a uniform distribution of items in the sequence. It turns out that the most interesting itemsets are quite large and can be found in a reasonable amount of time. These results were not surprising, as biologists confirmed that items were distributed in such a way. To double-check, we once again created a synthetic dataset, 375 items long, where items were distributed randomly, using their frequency in the DNA dataset to determine their probability of occurrence. Figure 1.8 (b) shows that the results obtained on this dataset are very similar to those in Figure 1.8 (a).
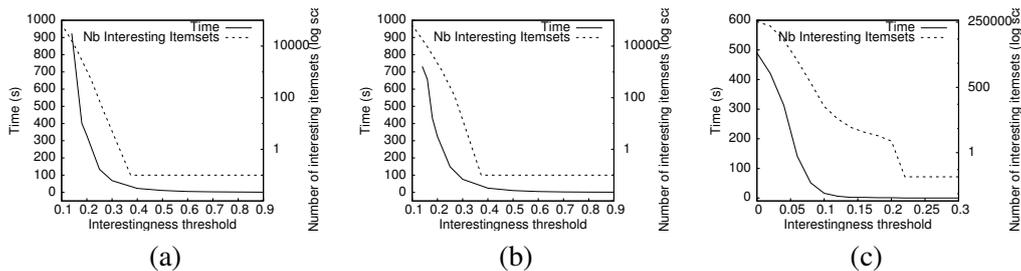
Figure 1.8: Execution time in seconds and number of extracted episodes (log-scale) with varying *min_interest*: (a) DNA dataset; (b) Random DNA-like dataset; (c) MUSIC dataset.

Our second dataset consisted of music notes, collected from one track of a midi-file. We used the notes of the song Abide With Me[2], having first converted the midi-file into a text file, and then pruned all other elements (pitch, channel, velocity, etc.[3]) from the file. This left us with a sequence of 400 notes, made of 18 distinct notes. Figure 1.8 (c) shows that no interesting itemsets were found using high interesting thresholds, but what was encouraging was the fact that the execution time was minimal. We found the first three interesting itemsets at the level of 0.2: one consisting of four notes, one of five, and the third was a singleton, which was also included in the other two sets. This indicated that this note appeared most often in the sequence, and that the other four notes always appeared relatively close to it (certainly closer than the other 13 notes). A larger number of itemsets was found to be interesting as we lowered the threshold further, growing consistently even for small decreases in the threshold.

---

[1]http://www.expasy.org/cgi-bin/sprot-ft-details.pl?P07327@SEQUENCE@2@375

[2]http://www.tc.umn.edu/~sorem002/hymn_midi.html

[3]http://www.fourmilab.ch/webtools/midicsv/

1.6 Discussion

In this chapter, I presented the constraint-based pattern mining framework that makes possible to achieve the exhaustive extraction of the whole set of patterns by exploiting the constraint properties to reduce the number of generated patterns. Several constraint properties, that have been identified as useful in this framework, were studied. Then I presented the formal concept pattern domain, its associated constraints and an algorithm to solve it. The principles of this algorithm can be abstracted so as to extract other pattern domains whose constraints are either monotone, anti-monotone or piecewise monotone and anti-monotone. Its foundations are the reducing of the search space either by enumerating elements, or by using the constraints. The constraints are applied (1) to determine as soon as possible that the candidate set is empty and (2) to certify that some elements must belong or be excluded to every pattern of the search space. Two case studies were detailed to illustrate the relevance of this algorithm and its usefulness in different contexts. Other use cases for attributed dynamic graph mining are given in Chapter 3.

The prospects of this work are numerous. It is still necessary to continue the study of constraint properties to understand their impact on the pattern search space as well as the data search space (Bonchi, Giannotti, Mazzanti, and Pedreschi 2005) with the prospect, for example, of developing efficient parallel algorithms (Negrevergne, Termier, Rousset, and Mehaut 2013). Another recent research topic is the study of set patterns (Crémilleux and Soulet 2008; Fürnkranz and Knobbe 2010; Raedt and Zimmermann 2007) and their associated constraints and algorithms. This very promising framework needs to be further investigated as new constraints on sets of patterns have to be identified to express the expected relationships between the retrieved local patterns. Finally, one of the main difficulties when using local pattern algorithms is how to fix the thresholds. This problem can be overcome for example by looking for skyline patterns (that is to say a set of incomparable patterns with respect to a domination relationship on the constraint thresholds (Soulet, Raïssi, Plantevit, and Crémilleux 2011)) or by statistically assessing the value of the measure and automatically deriving the constraint thresholds (Robardet, Scuturici, Plantevit, and Fraboulet 2013). This line of research is still to be followed.

2 — Relational dynamic graphs

While conducting our research to extend the formal framework of constraint-based pattern mining toward new pattern types, we have always considered real-world problems and try to extract knowledge in these contexts. To that end, in 2001, our data mining research team has established original collaborations with biologists to mine patterns in their gene expression data. The objective was to identify groups of genes whose expressions were similar in some biological contexts. These collaborations have resulted in the elicitation of hypothesis that have been validated by the involved biologists (Besson, Robardet, Boulicaut, and Rome 2005; Blachon, R. Pensa, Besson, Robardet, Boulicaut, and Gandrillon 2007; Leyritz et al. 2008). In 2006, we joined the Rhône Alpes Complex Systems Institute IXXI that was set up at that time. Our aim was to explore a new type of data, named as complex network, and to create new collaborations with researchers of graph theory and signal processing. We were motivated by the fact that complex network refers to a broad class of data – the systems composed by a large number of highly interconnected dynamical units – that covers lots of data generated by real systems. These systems can be modeled as relational graphs whose vertices represent dynamical units, and whose edges stand for interactions between them.

An important amount of complex network studies focused on the comparison of the topology of different real-world static networks and produced a series of unexpected results (see Boccaletti, Latora, Moreno, Chavez, and Hwang 2006 for a review on these works). They started by defining graph properties that can be used to characterize the topology of real networks and studied their statistical behavior on real-world complex networks. In a rather surprising manner, it appeared that some of these properties behaved the same way on many networks. The most cited result is probably the fact that the degree of a vertex – the number of edges that connect it to other vertices – has a distribution that significantly deviates from the Poisson distribution expected for a random graph and, in many cases, exhibits a power law tail (scale-free property, Barabási and Albert 1999). It has also been shown that real networks tend to have a degree assortativity, that is to say vertices tend to be linked to vertices with similar degree (degree correlation property, Echenique, Gómez-Gardeñes, Moreno, and Vázquez 2005) and to have relatively short paths between any two vertices (small-world property, Watts and Strogatz 1998). These properties have been used to model complex systems that can mimic real-world evolutionary mechanisms.

However, the way complex networks evolve, that is to say the way new vertices and edges appear while some old ones disappear, has been understudied and it appears crucial to better

understand these mechanisms. Their analysis brings new knowledge on the common rules that may govern the transformation of the network, and makes possible to generate random evolving networks that can be used for simulation purposes. In this context we conducted two in-depth analysis of mobility networks that are reported in the two following sections: In Scherrer, Borgnat, Fleury, Guillaume, and Robardet 2008, we proposed a framework for the study of wireless sensor mobility networks that addresses the description, analysis and the simulation of networks using techniques from signal processing, graph theory and data mining; In Borgnat, Robardet, Abry, Flandrin, Rouquier, and Tremblay 2013, we presented a joint analysis in space and time of the Vélo'v bicycle sharing system of Lyon based on the adaptation to the case of dynamical networks of community detection methods.

2.1 Description and simulation of wireless sensor mobility networks

While most of real-life complex networks are inherently dynamic, with vertices and edges appearing or disappearing through time, the dynamic of these networks was less studied and there is a strong need for dynamic network models in order to sustain fundamental analysis, and in the context of sensor networks, support protocol performance evaluations. In Scherrer, Borgnat, Fleury, Guillaume, and Robardet 2008, we proposed a novel framework for the study of dynamic mobility networks. This framework consists in first characterizing dynamic graphs by considering graph properties as function of time (see Section 2.1.2), such as the number of edges or the average degree, so as to give an empirical statistical signature of the dynamics. Global indicators of the dynamics are also presented (see Section 2.1.3). Based on the appearance or disappearance of edges, they do not simply relate to independent measures in the succession of static graphs of the dynamic network, but account for its evolution through time. From those observations, we design simple yet very accurate models (see Section 2.1.4) that generate random mobility graphs with similar temporal behavior as the one observed in experimental data.

2.1.1 The data

We study wireless sensor mobility networks, in which vertices are human beings and edges stand for the ability of a wireless communication to take place between them. This who-is-near-whom network evolves every time users move and communication services (such as the spread of any information) will deeply rely on the characteristics of the underlying network. More precisely, we analyze two mobility networks based on sensor measurements: The IMOTE data set (Hui, Chaintreau, Scott, Gass, Crowcroft, and Diot 2005), that has been collected during the Infocom 2005 conference – 41 participants kept a sensor during nearly 3 days – and the MIT experimental data set (Eagle and Pentland 2006) that is made of records from Bluetooth cell-phones contacts between 100 MIT students during 9 months.

2.1.2 Graph properties as function of time

The most natural way to describe the dynamics of a complex network is to study the evolution of static properties through time. At a given time step, a snapshot of the interactions that exist in the data set is modeled by a graph $G_t = (V, E_t)$ with $t \in \mathbb{N}$, the time index, constructed as follows: An edge $\{u, v\} \in E_t$ exists if the edge $\{u, v\}$ is present in the data at time t. Here, the set of vertices V does not change in time and we are interested in the dynamical aspect of the vertex interactions. The sampling period of G_t is 1 second for all analysis.

Static networks have been widely studied and a lot a simple parameters are available to describe a network: The number of connected vertices $V(t)$, the number of edges $E(t)$, the number of triangles $T(t)$, the number of connected components (excluding isolated vertices) $N_c(t)$, the average degree of connected vertices $D(t)$, the number of edge creations $E_\oplus(t) =$

Properties		IMOTE			MIT		
		Mean	Std. Dev.	Corr. Time (s)	Mean	Std. Dev.	Corr. Time (s)
#Active links	$E(t)$	21.9	12.4	5200	13	17.7	16800
#Connected vertices	$V(t)$	19.9	4.7	7400	12.3	11.6	17500
Avg degree	$D(t)$	2.1	0.8	3600	1.5	0.8	7300
#CC	$N_c(t)$	4.8	2.1	5600	3.7	2.7	5200
# Triangles	$T(t)$	6.9	8.3	4700	6.6	19.8	5500
Edge creation	$E_{\oplus}(t)$	0.15	0.55	680	0.005	0.11	30
Edge deletion	$E_{\ominus}(t)$	0.15	0.55	680	0.004	0.08	260

Table 2.1: Graph properties: Mean and Standard Deviation of PDF, Correlation Times.

$|\{e \in E_t, e \notin E_{t-1}\}|$ and the number of edge deletions $E_{\ominus}(t) = |\{e \in E_{t-1}, e \notin E_t\}|$.

For each time series, the probability distribution function is estimated over the duration of the observation, using empirical histograms of the data. The various estimated probability distributions obtained are not heavy tailed, meaning that the variability is not very large and the standard deviation is a good measurement of the variability. The mean and the standard deviation of the distributions are reported in Table 2.1 for a typical day of the IMOTE data set and a typical week of the MIT data.

The temporal evolution of a property $X(t)$ can be characterized by its correlation time (Abarbanel 1996). It is estimated as $C_X(\tau) = <X(t+\tau)X(t)>_t - (<X(t)>_t)^2$, where $<\cdot>_t$ is the mean over time. The correlation time is then defined as the first time where the function $C_X(\tau)$ goes to zero (this always happens due to the summation rule of empirical C_X). It quantifies the "memory" of the series: The longer it is, the greater is the persistence of fluctuations in the data. The correlation times (see Table 2.1) of E, and V are rather large and all of the same magnitude: $\sim$ 1h45 for IMOTE and $\sim$ 4h45 for MIT. The properties D, N_c and T have comparable correlation times. This suggests that these properties evolve under a common cause. However, the correlation time of the edge creation and deletion properties is really much smaller ($\sim$ 11 min. for IMOTE and $\sim$ 2 min. for MIT) than the correlation time of all other characteristics. Therefore, these properties can almost be considered memory-less.

We also consider contact and inter-contact duration distributions, that are dynamic characteristics interesting for mobility networks. The contact duration is the time during which two vertices remain directly and continuously adjacent. The inter-contact duration is the duration between two periods of contact for two vertices. As observed in Hui, Chaintreau, Scott, Gass, Crowcroft, and Diot 2005 for the IMOTE data, both distributions have a tail that can be modeled by a power law, so that the complementary cumulative distribution functions (CCDF) of contact or inter-contact durations X (seen as a random variable) follows: $P[X > x] \underset{x\to\infty}{\sim} cx^{-\alpha}$. For $\alpha > 2$, the associated random variable X has a finite mean and a finite variance. For $\alpha < 2$, X has an infinite variance and is said *heavy tailed*. Moreover, if $\alpha < 1$, X has both an infinite mean and an infinite variance. When the variance is infinite, the tail dominates the behavior of the variable, especially it causes large events much more frequently than for non-heavy tailed distributions. Therefore high variability in the data is sometimes explained by heavy tails, as opposed to lower variability which appears for instance with exponentially decaying tails.

Figure 2.1 shows the CCDF of contact and inter-contact durations and the fitted power-law distributions (mean and estimated α are reported in captions), respectively for the IMOTE and MIT data sets. The fits show relevancy of the power-law behavior for both contact and inter-contact duration distributions over a wide range of scales. As already discussed in Hui, Chaintreau, Scott, Gass, Crowcroft, and Diot 2005 for the IMOTE data, the heavy-tailed nature of these distributions seems to be an ubiquitous property of dynamic mobility networks. For both data sets, inter-contact duration distributions have an α lower than 1, meaning very strong variability due to long periods of lack of contact for some vertices, whereas the distribution of

IMOTE

Mean: 140 - $\alpha = 1.66$ Mean: 3680 - $\alpha = 0.60$

MIT

Mean: 3270 - $\alpha = 2.4$ Mean: 148700 - $\alpha = 0.49$
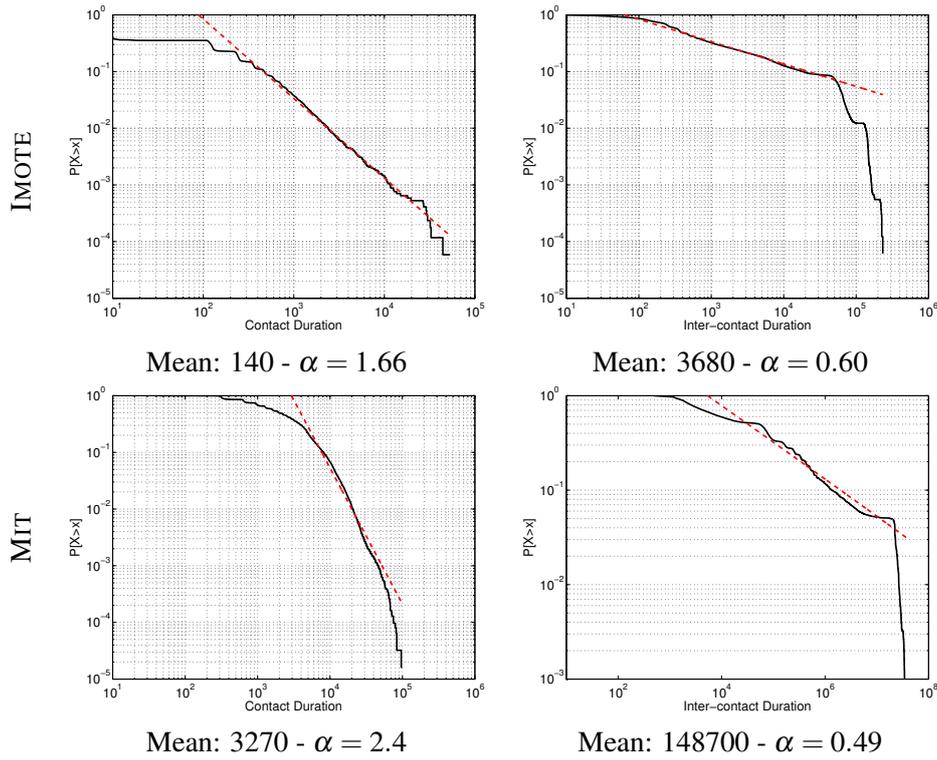
Figure 2.1: Contact (left) and Inter-contact (right) duration distributions (CCDF).

contact durations is less heavy-tailed (for MIT it is actually not heavy-tailed at all, with $\alpha = 2.4$).

Let us emphasize as a conclusion to this subsection, that, in the dynamic mobility networks studied, (1) the edges creation and deletion processes are memory-less, and (2) the contact and inter-contact duration distributions can be modeled by a power-law. These properties are used in our simulation algorithms presented in Section 2.1.4.

2.1.3 Analyzing dynamic properties

Considering the evolution as a sequence of snapshots is an efficient and simple approach in many cases but some properties cannot be directly observed in this framework. This is the case of global properties that can be used to characterize the dynamics of a graph as a whole. To that end, one has to consider the evolution of the network from one time step to the next. We study hereafter the stability of connected components (and hence of the information paths between vertices), the proportion of creation of triangles observed as well as the communities embedded in the network.

Stability of Connected Components

A connected component (CC) is a maximal sub-graph such that a path exists between every pair of vertices. We consider the *total lifetime* of a connected component *c* defined as the number of time steps for which *c* exists, and the *number of appearances* of *c* defined as the number of time steps *t* for which this CC is absent at *t* and present at *t* + 1. Figures 2.2 (left) and (right) display the distributions of the total lifetime and the number of appearances of all CC for IMOTE. These plots exhibit a strong heterogeneity for both parameters. While more than 52% of the CC exist only during one time step, some of them exist during a quarter to a third of the whole time.

To give a better insight, Figure 2.3 (left) shows a joint distribution of the number of vertices
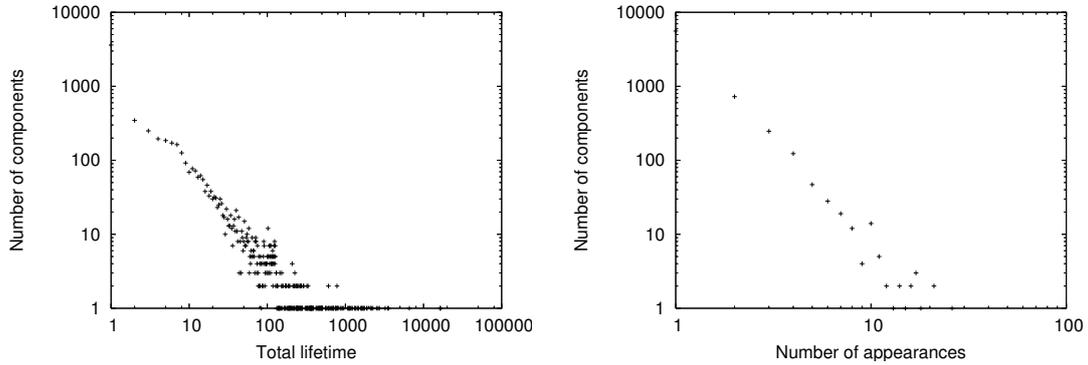
Figure 2.2: Distribution of the total lifetime (left), number of appearances (right) for all CC of IMOTE.

and total lifetime for all CC of IMOTE. The absence of stability of large CC is striking: There is no CC with more than 8 vertices (while 67% CC have more than 8 vertices) which have a lifetime greater than 100 seconds. The more edges and vertices a CC contains the more potential modifications may happen, which explains in part the curves. However, we could have expected that such sets would be stable for longer periods. Figure 2.3 (right) presents similar results for the joint distributions of the number of vertices and number of appearances for all CC of IMOTE. Again the CC which reappear regularly are only small components. The plots might lead us to believe that some large CC appear regularly more than once. However, if we admit that a component reappears only if it has disappeared for more than five minutes, then no CC of size greater than 6 appears more than once (representing 73% of all CC). This means that most CC reappear very soon after they have disappeared, which is a direct consequence of edges and vertices flickering.
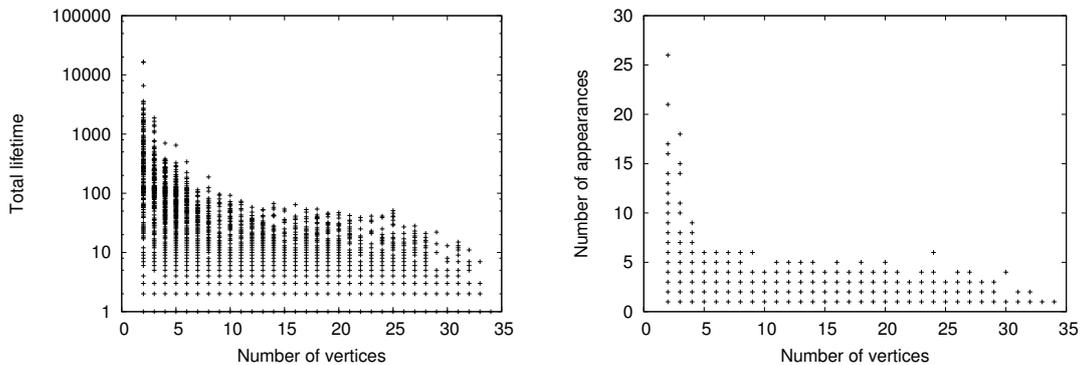
Figure 2.3: Joint distribution of the number of vertices and total lifetime (left), and joint distribution of the number of vertices and number of appearances (right) for all CC of IMOTE.

Edge appearance and triangle creations

The existence and persistence of connected components is generally associated with a rather large number of triangles in the graph. Therefore, it is interesting to evaluate the proportion of edges whose appearance creates triangles. To that end, we count the proportion of edge creations that leads to an increase of the number of triangles in the graph ($P_{+/tri+}$) or does not change this amount of triangles ($P_{+/tri=}$). Let further $f_{+/tri+}$ (resp. $f_{+/tri=}$) be the average proportion of inactive edges that would create (resp. would not create) a triangle if activated.

These proportions are given in Table 2.2 (in percentage) for IMOTE data, MIT data and for a simple random dynamical graph – with independent contact and inter-contact distributions distributed as a power-law. One can see that for both data sets, around 40% of edge creations increase the number of triangles in the graph, which is a fairly large proportion when compared to a simple random dynamical graph ($\sim$ 10%). The proportion of inactive edges that would create a triangle is very low for both experimental data sets as well as for the simple random graph. This emphasizes the fact that this is not because more edges can create triangles that the proportion $P_{+/tri+}$ is higher in experimental data sets: It is on the contrary an intrinsic property of the dynamics.

	$P_{+/tri+}$	$P_{+/tri=}$	$f_{+/tri+}$	$f_{+/tri=}$
IMOTE	44 %	56 %	6 %	94 %
MIT	40 %	60 %	7 %	93 %
RANDOM	10 %	90 %	5 %	95 %

Table 2.2: Proportion of edge creations that adds a new triangle or not (P), and proportion of inactive edges that, if created, would add a triangle, or not (f).

Qualitative summary of the dynamic interactions

To account for the dynamics of the network, it is also important to provide a qualitative summary of the interactions that durably take place in the network. This can be achieved by isolating "communities" that are commonly considered as groups of individuals with *many* and *dense* interactions between each other over a (not necessarily continuous) *long* period of time. In other words, a community can be seen as a dense connected sub-graph with many edges that appears in a large number of time steps. We propose to identify such communities using the constraint-based pattern mining framework (see Chapter 1). The procedure consists of two steps whose execution requires to set several parameters. In the first step, it gathers information on groups of edges over time by computing sub-graphs with at least σ edges, a density greater than δ and that appear in at least τ time steps of the network. In the second step, sub-graphs that concern similar individuals and time steps (the difference between two time steps is at most Δ) are merged to obtain important and established groups of persons that durably interact. This enables us to take into account the variability (edge flickering) observed in the experimental measurements. Finally, the dynamic trajectories of the persons are built by considering, for each individual, the communities she belongs to and ordering them with respect to time to obtain the trajectories.

Computing large, frequent and dense connected sub-graphs: Let $\varphi = (N, L)$ be a partial sub-graph of $G_t = (V, E_t)$, denoted hereafter $\varphi \subseteq G_t$, if $L \subseteq E_t$ and N is the set of vertices that appear as endpoint of at least one edge of L. The sub-graphs can be partially ordered using the inclusion relation of their edge sets: $\varphi_1 = (N_1, L_1) \preceq \varphi_2 = (N_2, L_2)$ iff $L_1 \subseteq L_2$. The sub-graphs of interest in $\mathscr{D} = \{G_t, t \in \mathbb{N}\}$ are defined thanks to four constraints. To be sufficiently large, these sub-graphs must be made of at least σ edges:

$$\mathscr{C}_{large}(\varphi, \mathscr{D}) \equiv |L| \geq \sigma$$

To be sufficiently frequent, they must be included in at least τ time steps:

$$\mathscr{C}_{frequent}(\varphi, \mathscr{D}) \equiv |\{t, \varphi \subseteq G_t\}| \geq \tau$$

They are also expected to be connected:

$$\mathscr{C}_{connected}(\varphi, \mathscr{D}) \equiv \forall w_0, w_k \in N, \exists (w_0, w_1)(w_1, w_2) \cdots (w_{k-1}, w_k) \text{ with } (w_i, w_{i+1}) \in L$$

Finally, these induce sub-graphs have to be dense:

$$\mathscr{C}_{dense}(\varphi, \mathscr{D}) \equiv \frac{2|L|}{|N|(|N|-1)} \geq \delta$$

We can observe that these constraints satisfy the following properties.

Property 2.1 $\mathscr{C}_{large}(\varphi, \mathscr{D})$ is monotone with respect to $\preceq$.

Proof. Let $\varphi_1 = (N_1, L_1) \preceq \varphi_2 = (N_2, L_2)$. If $\mathscr{C}_{large}(\varphi_1, \mathscr{D})$ is satisfied then $|L_1| \geq \sigma$ and as $L_1 \subseteq L_2$, we have $|L_2| \geq \sigma$. Thus, $\mathscr{C}_{large}(\varphi_2, \mathscr{D})$ is also satisfied.   ∎

Property 2.2 $\mathscr{C}_{frequent}(\varphi, \mathscr{D})$ is anti-monotone with respect to $\preceq$.

Proof. Let $\varphi_1 = (N_1, L_1) \preceq \varphi_2 = (N_2, L_2)$. If $\mathscr{C}_{frequent}(\varphi_2, \mathscr{D})$ is satisfied then $|\{t, \varphi_2 \subseteq G_t\}| \geq \tau$. As $\varphi_2 \subseteq G_t$ implies $\varphi_1 \subseteq G_t$, we have $|\{t, \varphi_1 \subseteq G_t\}| \geq \tau$ and $\mathscr{C}_{frequent}(\varphi_1, \mathscr{D})$ is also satisfied.   ∎

Property 2.3 $\mathscr{C}_{connected}(\varphi, \mathscr{D})$ is loose anti-monotone with respect to $\preceq$.

Proof. Let $\varphi = (N, L)$ be a sub-graph that satisfies $\mathscr{C}_{connected}(\varphi, \mathscr{D})$. There exists an edge $(u, v) \in L$ such that $\mathscr{C}_{connected}(\varphi \setminus (u, v), \mathscr{D})$ is also satisfied. Indeed,
- either φ contains a cycle, and removing (u, v), an edge of this cycle, does not disconnect the sub-graph – there is still a path from u to v,
- or φ is a tree and thus it contains at least two vertices of degree 1 (the extremities of a path of maximum length in the tree). Removing an edge whose one of its endpoints is of degree 1 will decrease the number of vertices of the sub-graph but not disconnect it.

          ∎

$\mathscr{C}_{dense}$ constraint does not satisfy appropriate properties when considering partial sub-graphs, that is to say sub-graphs enumerated by their set of edges. Indeed, this constraint is known to be loose anti-monotone when considering the sub-graph induced by a subset of vertices (see Uno 2010 for a proof), but there is no such property when enumerating them by their set of edges.

Therefore, we use the generic algorithm presented in Section 1.4 to compute such patterns, only exploiting the three first constraints during the enumeration process. The two first ones can be directly implemented in the algorithm. The third one requires to specify the enumeration process of the edges to only list connected partial sub-graphs: Starting from a partial sub-graph φ, the next edge to be enumerated by the algorithm is one whose at least one of its endpoints belongs to φ. If no such edge exists, no more connected sub-graphs can be generated from φ and the enumeration stops. This enumeration process is complete since, as stated by property 2.3, for any connected partial sub-graph there exists a connected sub-graph with an edge less. Finally, $\mathscr{C}_{dense}$ constraint is checked in a post-treatment.

Identifying communities and individual trajectories: The second step consists in merging the obtained sub-graphs that share similar vertices and time steps. Two sub-graphs $\varphi_1 = (N_1, L_1)$ and $\varphi_2 = (N_2, L_2)$, whose supports are respectively T_1 and T_2, are merged with respect to Δ if $N_1 \subseteq N_2$, and $\forall t \in T_1 \setminus T_2, \exists t_0 \in T_2, |t - t_0| < \Delta$. The obtained sub-graph $\varphi = (N_2, L_1 \cup L_2)$ is associated with time steps $T_1 \cup T_2$ and is considered as a community. Individual trajectories are built from these communities.

	τ	σ	# sub-graphs	Avg. # vertices	Avg. # edges.	Avg. # time steps
IMOTE	7	6	27507	7.9	8.3	8.2
MIT	10	14	30144	10.5	12.8	18.1

Table 2.3: Algorithm parameters and frequent connected sub-graphs properties.

Results obtained on IMOTE and MIT data sets: Table 2.3 reports the number and size of large, frequent and connected sub-graphs that have been extracted, as well as the parameters used. Among these sub-graphs, only those with a density greater than $\delta = 0.8$ are kept. This gives us 138 dense connected sub-graphs for IMOTE and 1226 for MIT. As these patterns overlap, we use the second step using $\Delta = 15$ (30 min. in real time) to obtain communities, *i.e.* distinct dense connected sub-graphs and their associated time steps. It results that IMOTE data set is structured by 14 communities (see Figure 2.4), whereas there are 9 communities on the MIT data set.

From these communities we derive the trajectories of the individuals. The trajectory of each individual in the communities of IMOTE data is displayed in Figure 2.4. Boxes represent individuals entering a group and edges are labeled by the individual number when he/she goes from one group to another. Notice that the graph is oriented: An arc (u,v) represents individuals moving at least once in the data from group u to group v. For example, individual 8 initially belongs to group 13, he/she further moves into group 6, and finally enters group 7. Results on MIT are similar and are not reported here. Note that these techniques use exhaustive methods and algorithms but still require a supervisor to fix several parameters $(\sigma, \delta, \tau, \Delta)$ to drive the graph structural exploration.
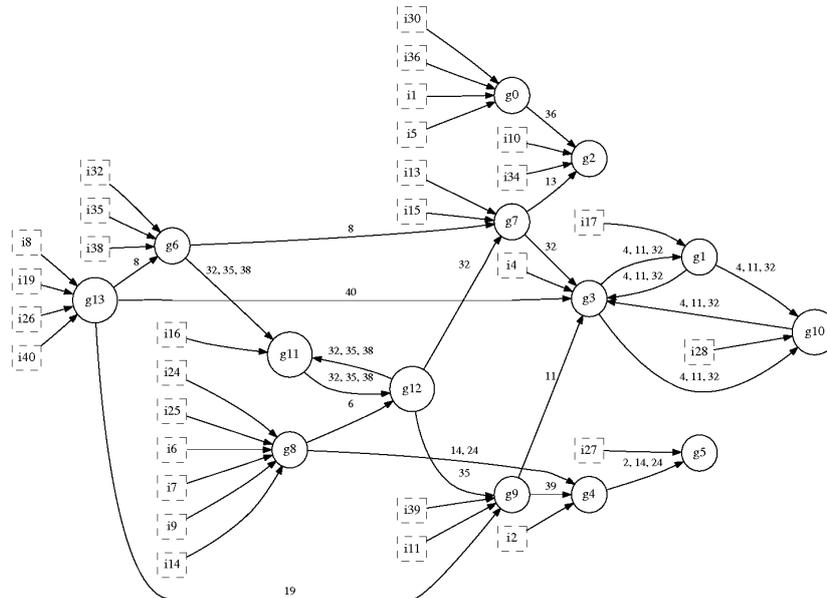
Figure 2.4: Individual trajectories in groups ordered by time. i_x (boxes) are individuals while g_x (circles) denotes social groups.

Let us emphasize as a conclusion to this subsection, that the dynamic mobility networks studied display non-trivial global properties: The connected components have all the more short lifetime since they are large; Around 40% of edge creations increase the number of triangles in the graph; From these dynamic graphs, one can identify ten groups into which individuals are moving. These properties are to be central ones to keep in modeling those networks. In the following section, we propose simulation algorithms that generate data with similar properties.

2.1.4 Simulation of Dynamic Graphs

From the observations detailed in Sections 2.1.2 and 2.1.3, we propose several generic random dynamic models that make possible to generate random dynamic graphs which have a behavior similar to the one observed in experimental data sets. These algorithms simulate the edge creation and deletion processes by considering several input data: The number of vertices, the contact and inter-contact duration distributions, the connected component number distribution or the dynamics of triangle creation. Their ability to simulate credible data is assessed by comparing the previously mentioned properties on both original and simulated data.

Simulation algorithm

The simulation is based on a transition model with Markovian property. For each time step and for each edge independently, each edge e will change its state (active or inactive) with transition probability $P_{tr}(e, G_t)$. The corresponding algorithm is displayed as Algorithm 3. The transition probability depends only on the state of the network, in particular on the duration $\tau(e)$ since the link e has last changed its status (up-time if the edge is active, down-time if it is inactive). The assumptions that the transition probabilities are independent from one time step to another hold because the processes of edge creation and deletion were found to have a correlation time much smaller than any other property of the graph. Moreover, we can consider each edge independently since edge correlation coefficients (in time) were found very low[1]. Four models for the transition probabilities are proposed hereafter, that combine the contact and inter-contact duration distributions, as well as the number of connected component distribution, and the triangle creation process.

Algorithm 3 Simulation algorithm

Require: Simulation time
Ensure: Random Dynamic Graph
 1: **for all** Simulation Time Step t **do**
 2:　　**for all** Edge e **do**
 3:　　　$P_{tr}(e, G_t)$ = TransitionProbability(e) given the state G_t
 4:　　　p_r = Uniform(0,1)
 5:　　　**if** $(p_r \leq P_{tr}(e))$ **then**
 6:　　　　ChangeState(e)
 7:　　　**end if**
 8:　　**end for**
 9: **end for**

Model $\mathcal{A}$: A basic point of the dynamics was expressed as heavy-tailed distributions for contact and inter-contact durations (referred to as P_{ON} and P_{OFF} respectively), as discussed in Section 2.1.2. Supposing them to be stationary distributions for the system, we derive the transition probabilities of the edge, depending on $\tau(e)$ (time since the edge is in the state). Let $P_+(\tau)$ be the probability that one edge that was OFF (*i.e.*, inactive) since τ ($\tau \geq 1$) is activated (going from OFF to ON), at a given time. Similarly, let $P_-(\tau)$ be the probability that one edge that was ON (*i.e.*, active) for a time τ ($\tau \geq 1$) is deleted (going from ON to OFF). The probability that a contact will last for τ time steps can be computed as the probability that the edge disappeared after τ multiplied by the probability that the edge did not disappear in the preceding $\tau - 1$ time steps. It can be expressed as $P_{ON}(\tau) = P_-(\tau) \times \prod_{i=1}^{\tau-1}(1 - P_-(i))$. One can

---

[1]Edge creation and deletion processes ($E_\oplus(t)$ and $E_\ominus(t)$) remain mostly uncorrelated with all other properties with correlation coefficients always below 0.2.

then invert this relation and compute $P_-(\tau)$ (resp. $P_+(\tau)$) recursively as:

$$P_-(\tau) = \frac{P_{ON}(\tau)}{\prod_{i=1}^{\tau-1}(1-P_-(i))}, \quad \tau \geq 2, \quad P_-(1) = P_{ON}(1)$$

$$P_+(\tau) = \frac{P_{OFF}(t)}{\prod_{i=1}^{\tau-1}(1-P_+(i))}, \quad \tau \geq 2, \quad P_+(1) = P_{OFF}(1)$$

In this model, named $\mathcal{A}$, the empirical contact and inter-contact duration distribution is only imposed and the transition probability is therefore:

$$P_{tr}(e, G_t) = \begin{cases} P_-((\tau(e)) \text{ with } \tau(e) \text{ the time since } e \text{ is active} \\ P_+((\tau(e)) \text{ with } \tau(e) \text{ the time since } e \text{ is inactive} \end{cases}$$

It is easy to check that all resulting sequences of graphs G_t will share on average the prescribed distribution of contact and inter-contact durations. As it was already shown in Fleury, Guillaume, Robardet, and Scherrer 2007, merely forcing the contact and inter-contact duration distributions is not sufficient to fully uncover the dynamics of an experimental data set. It is then worth introducing other elements in the transition probability.

Model $\mathcal{B}$: To explore the relevance of properties such as $E(t)$, $V(t)$, $N_C(t)$ and $D(t)$, the probability of transition is weighted by a probability of acceptance of the new state depending of the experimental distribution for a property of interest. This is implemented by Rejection Sampling (Robert and Casella 2004), based on a Metropolis-Hastings algorithm (Hastings 1970). The new proposed state, denoted $G_t' = \{G_t \text{ with state of } e \text{ changed}\}$, is accepted with probability $P_{RS}(G_t, G_t') = \min\left(1, \frac{F(x(G_t'))}{F(x(G_t))}\right)$ if F is the target PDF for the graph. The rationale of this procedure is to impose the averaged distributions on the simulated sequence of graphs.

From all the possible graph properties that can be simulated by our random dynamical graph generator, one graph property is emphasized for the sake of the clarity: The number of connected components. Because other properties were found highly correlated with one another, the behavior of the simulation models is similar with respect to them. We denote by $\mathcal{B}$ the model that imposes the distributions of contact / inter-contact durations and of the number of connected components:

$$P_{tr}(e, G_t) = P_{-/+}(\tau(e)) \cdot P_{RS}(G_t, G_t').$$

Models $\mathcal{A}_\omega$ and $\mathcal{B}_\omega$: The average proportion of edges creations that yield triangles is larger than for random graphs as discovered in Section 2.1.3. Therefore, to take this into account in the simulations, a weight is applied on the transition probability to reproduce the correct dynamical transition process concerning triangles (and not merely the stationary distribution of the number of triangles). Obviously, we do not want to change the mean probabilities of transition. Hence, the weights are chosen such that the mean probability (over all the inactive edges) is still $P_+(\tau)$. Using the same notations as in Section 2.1.3, the transition probabilities are corrected with the ratios $P_{+/tri+}/f_{+/tri+}$ for activation of edges that add a triangle, and $P_{+/tri=}/f_{+/tri=}$ for those that do not. This complies naturally with the fact that the averaged probabilities of creation are not changed. Nevertheless, it will give a higher transition probability to triangle-affecting transitions, like in IMOTE and MIT data sets.

We apply this weight on the two previous models and obtain two other simulation models:
- $\mathcal{A}_\omega$, with the following transition probability:

$$P_{tr}(e, G_t) = \begin{cases} P_+(\tau(e))\frac{P_{+/tri=}}{f_{+/tri=}} & \text{for edge creation without new triangle,} \\ P_+(\tau(e))\frac{P_{+/tri+}}{f_{+/tri+}} & \text{for edge creation with a new triangle.} \\ P_-(\tau(e)) & \text{for edge deletion.} \end{cases}$$

- and $\mathcal{B}_\omega$ with the following transition probability:

$$P_{tr}(e, G_t) = \begin{cases} P_+(\tau(e))P_{RS}(G_t, G'_t)\frac{P_{+/tri=}}{f_{+/tri=}} & \text{for edge creation without new triangle,} \\ P_+(\tau(e))P_{RS}(G_t, G'_t)\frac{P_{+/tri+}}{f_{+/tri+}} & \text{for edge creation with a new triangle.} \\ P_-(\tau(e))P_{RS}(G_t, G'_t) & \text{for edge deletion.} \end{cases}$$

All models are stationary in that the parameters are fixed for the full simulation. The simulations presented here are designed to reproduce the first day of conference in the IMOTE data set (this period corresponds to $0.55 \cdot 10^5$s to $1.0 \cdot 10^5$s), a period where the data appears conveniently stationary – even though the whole data set is not.

Simulation results

Models $\mathcal{A}$ and $\mathcal{B}$: Figure 2.5 (left) shows the number of links for IMOTE data and the first model $\mathcal{A}$. Obviously, as we are considering stationary models, they cannot account for peaks corresponding to lunches, breaks, etc. Simply the average number of links in both IMOTE and the models is the same. Figure 2.5 (middle and right) shows the contact and inter-contact duration distributions of both the original IMOTE data and these two models. The distributions are almost perfectly adjusted in all cases, showing that even when targeting a distribution using the rejection sampling step, the contact and inter-contact distributions are properly reproduced. This constitutes a basic validation of our simulation procedure.

—— Imote / o red Model $\mathcal{A}$ / * green Model $\mathcal{B}$
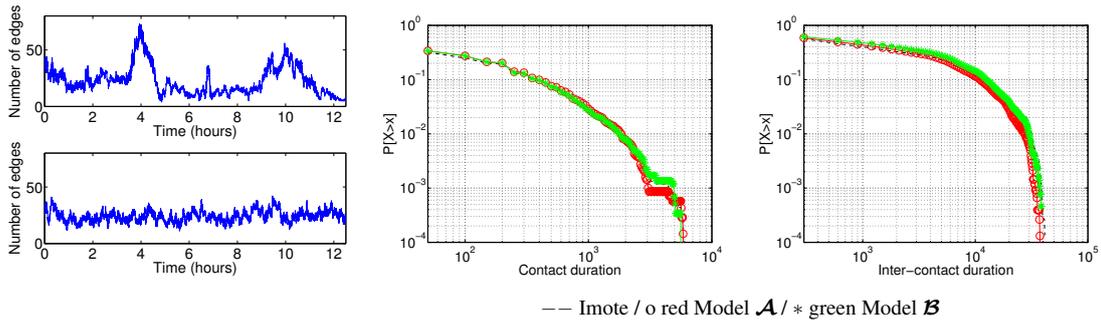
Figure 2.5: Number of edges for IMOTE data and model $\mathcal{A}$ (left), contact (middle) and Inter-contact (right) duration distributions (CCDF) for the models and IMOTE.

The distributions of $E(t)$, $V(t)$ and $N_c(t)$ are plotted in Figure 2.6. A first remark is that the sole contact and inter-contact duration distributions (model $\mathcal{A}$) dramatically fail to reproduce the properties. More precisely, the number of connected vertices is strongly over-estimated, the number of connected components is under-estimated, and so is the number of triangles. The non-stationarity in the IMOTE data introduces a much higher variance, yet it does not explain all the differences. Imposing the distribution of connected components (model $\mathcal{B}$) improves the accuracy of the simulation.

When more global characteristics such as the joint distribution of number of edges and vertices in connected components are estimated, the IMOTE data (shown on Figure 2.7 (left)) and the simulations of the models $\mathcal{A}$ and $\mathcal{B}$ (shown on Figure 2.7 (middle and right)) are much different. The connected components in the two models are much less dense, for a given number of vertices, the number of edges is often the minimal one (the dotted line on the plots), and does not vary much above this minimum. Neither the contact/inter-contact duration distributions nor the stationary distributions of standard graph properties manage to reproduce dense connected components as observed in both IMOTE and MIT data sets. The density of the connected components (the groups) is still underestimated in the previous models. Edges are spread uniformly in the graph, and consequently fail to create large and dense connected

−− Imote / o red Model $\mathcal{A}$ / ∗ green Model $\mathcal{B}$
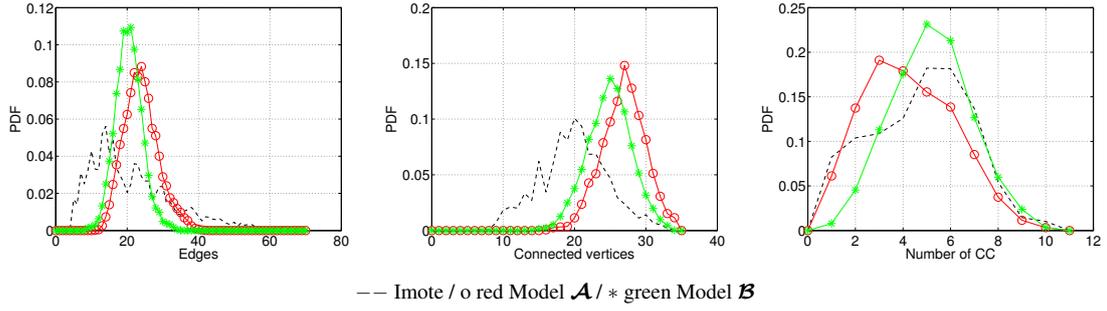
Figure 2.6: IMOTE: Probability distribution function for original data and the models.

components. We believe this is of major importance for communication protocol design and realistic models have to reproduce this property. The same remarks can be made for the joint distribution of the number of connected vertices and edges in the graph.
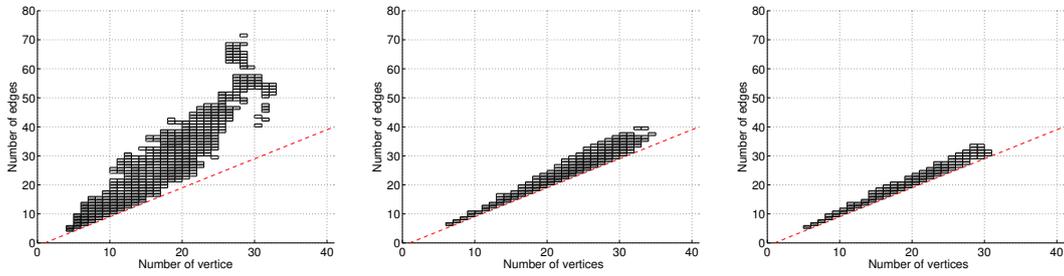
Figure 2.7: Joint distribution of the number of connected vertices and edges in connected components, for IMOTE (left), model $\mathcal{A}$ (middle) and model $\mathcal{B}$ (right).

Models $\mathcal{A}_\omega$ and $\mathcal{B}_\omega$: A first observation is that the introduction of the triangle creation probability does not have an impact on the contact and inter-contact duration distributions (not reported here). As imposed in the models, the probabilities $P_{+/tri+}$ and $P_{+/tri=}$ are much closer to experimental data, see Table 2.4. The fact that $P_{+/tri+}$ is slightly over-estimated is again due to an asymmetry in the transition probability among edges.

Data set	$P_{+/tri+}$	$P_{+/tri=}$
IMOTE	44 %	56 %
Models $\mathcal{A}$ and $\mathcal{B}$ (average)	5 %	95 %
Models $\mathcal{A}_\omega$ and $\mathcal{B}_\omega$ (average)	60 %	40 %

Table 2.4: Proportion of edges additions that creates a new triangle for the four models.

Figure 2.8 reports the joint probabilities of the number of connected vertices and edges in the graph as well as inside the connected components. As opposed to the models $\mathcal{A}$ and $\mathcal{B}$, the density of connected components is comparable to that of IMOTE data. The model, thanks to the introduction of dynamical characteristics, manages to generate more realistic simulations. This opens the track to improved models that match the important characteristics of dynamics of mobility networks.

To qualitatively assess our simulation model, we report on Figure 2.9 the trajectories of individuals among identified communities for model $\mathcal{B}_\omega$, using the same methodology as in Section 2.1.3. For models $\mathcal{A}$ and $\mathcal{B}$, it is impossible to identify any communities satisfying
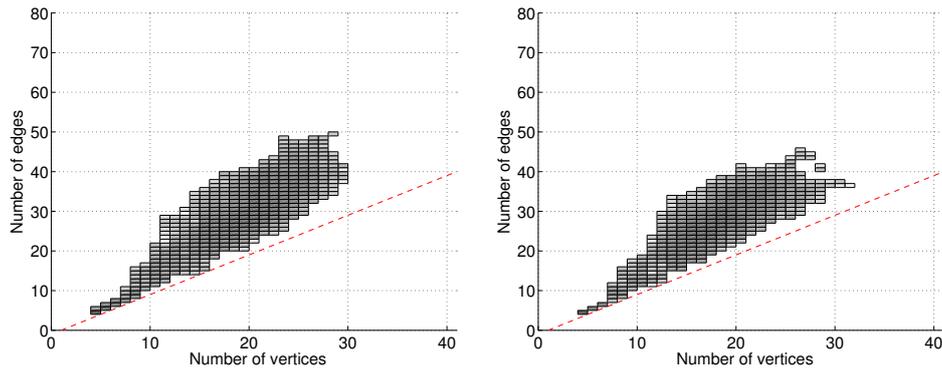
Figure 2.8: IMOTE: Joint distribution of the number of connected vertices and edges in connected components, for models $\mathcal{A}_\omega$ (left) and $\mathcal{B}_\omega$ (right).

the density and temporal support constraints. Because the number of frequent connected subgraphs is larger on the simulated data than on the original ones, the parameters σ and τ are increased from respectively 6 and 7 on the IMOTE data, to 9 and 10 on the simulated ones (see Section 2.1.3), so as to keep a reasonable number of communities. Trajectories comparable to the ones computed on IMOTE data are found here, once again emphasizing that models $\mathcal{A}_\omega$ and $\mathcal{B}_\omega$ are more realistic than the other ones.
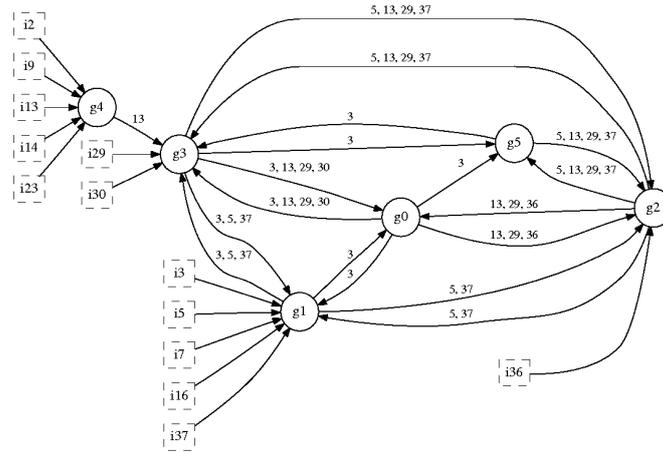
Figure 2.9: Trajectories of individuals among communities extracted from $\mathcal{B}_\omega$.

In parallel to this theoretical study on publicly available data, we have worked on high-value data: footprints of each rental of the Vélo'v bicycle sharing system. These data includes all information on the Vélo'v rentals, that is to say, the full information of 2 % of all trips in the city whatever the means of transport used. The following Section presents some results we obtained from these data.

2.2 Vélo'v bicycle sharing system viewed as a complex network

As part of IXXI Institute, a group of researchers from various disciplines was formed in order to study Lyon's shared bicycle system called Vélo'v. This public transportation system provides digital footprints of all the movements made using this system and makes possible its study using a dynamic complex network point of view. The Vélo'v program is deployed in Lyon since

May 2005. It consists now of 4000 bicycles (also called Vélo'v) that can be hired at any of the 340 stations, spread all over the city, and returned back later at any other station. In contrast to old-fashioned rental systems, the rental operations are fully automated: The stations are in the street and can be accessed at anytime, and the rentals are made through a digital terminal at the station with a credit card to obtain a short-term registration card, or through a year-long subscription system. Thanks to JCDecaux – Cyclocity and the Grand Lyon, we have access to the anonymized version of the trips made with Vélo'v over more than two years (from May 2005 to the end of 2007). During this period, there were more than 13 millions bicycle trips. The dataset consists of a log of all the rentals, with the station and the time of departure, and the destination station and the time of arrival.

Borgnat, Robardet, Abry, Flandrin, Rouquier, and Tremblay 2013 presents a joint analysis in space and time of Vélo'v system, in order to discover the main patterns of use along these two dimensions and obtain some insight on the dynamics of people's moves. This analysis is based on previous results that are recalled in Section 2.2.1. Section 2.2.2 details how Vélo'v bicycle trips can be accumulated to form a relational dynamic graph. Finally, network stations are aggregated on the basis of (1) the number of bicycles they exchange during the week-days or the week-ends (Section 2.2.3), or (2) the similarity of their usage pattern (Section 2.2.4).

2.2.1 Global features of Vélo'v system

A basic feature of transportation system studies is to discover its time pattern of use: When is it really used? What are the peak hours? Is it a means of transportation for the ordinary week-days or the week-ends? Using periodic averaging over the week combined to detrending of the nonstationary behavior, we are able to estimate the mean pattern of total rentals along the week, as shown in Figure 2.10 (Borgnat, Abry, Flandrin, Robardet, Rouquier, and Fleury 2011; Borgnat, Abry, Flandrin, and Rouquier 2009). It reveals that Vélo'v is used first for ordinary transport on working days. Its activity peaks are during the morning, the lunch time and the evening and this is characteristic of a system used to go to work and then to come back, with the lunch break in the middle. All the week-days are similar in that aspect, as shown on Figure 2.10 (b). However, a second type of use exists during the week-ends, as seen on this second plot: The peaks are less sharp and more spread out around noon and during the afternoon. This is compatible with leisure activities. Finally, one can see just after midnight each day a small bump that can be related to the closure of the public transports right after midnight (this even causes local maxima on Friday and Saturday nights). This type of pattern is reminiscent to uses of ordinary public transports. An exception is the nocturnal activity when public transports are closed.

The prediction of the number of rentals at a given day and time was addressed in Borgnat, Abry, Flandrin, Robardet, Rouquier, and Fleury 2011; Borgnat, Abry, Flandrin, and Rouquier 2009 using statistical time-series analysis. The global number of rentals, summed over the city, can be predicted on an hourly basis if one takes into account several important factors: The weather (temperature and rain), the holiday periods, and the existence of a correlation over one hour. The first two features (temperature and rain) account for most of the nonstationary evolution over the year of the average rentals. When zooming in at finer time scales, the one-hour correlation reflects that one decides to use a bicycle depending on the conditions seen during the previous minutes, and a rain condition affects the decision on this hourly scale – as it could be expected.

Finally, several empirical studies of Vélo'v data revealed various features of this system, such as the most frequent paths taken by these bicycles, the advantages of using a bicycle as compared to a car (Jensen, Rouquier, Ovtracht, and Robardet 2010), the distribution of durations and lengths of the trips (Borgnat, Abry, Flandrin, Robardet, Rouquier, and Fleury 2011), or the

(a) (b)
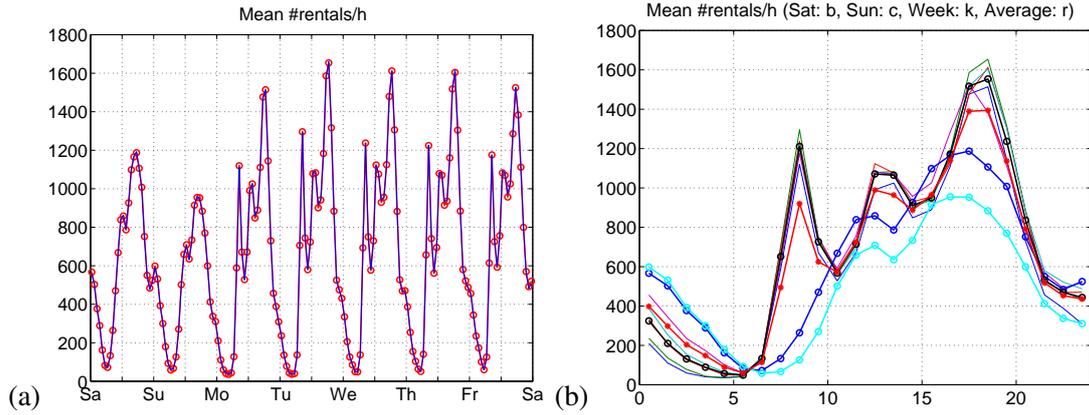
Figure 2.10: Number of rentals made per hour and per day of week, summed over all the Vélo'v stations: (a) pattern for the whole week; (b) superposition of individual day patterns in the week. The average over the 5 ordinary week-days from Monday to Friday is the thick black curve with circles. The five curves in thin lines that look alike are each for one week-day: Monday in blue; Tuesday in dark green; Wednesday in red; Thursday in cyan; Friday in purple. Saturday (thick blue curve with circles) and Sunday (thick cyan curve with circles) reveal a different structure, especially for the peak hours. The average over all days is the thick red curves and is dominated by the week-days.

usual speed of the bikers depending on the time of the day (Borgnat, Abry, Flandrin, Robardet, Rouquier, and Fleury 2011; Jensen, Rouquier, Ovtracht, and Robardet 2010). Note that these properties are generally heterogeneous because their distributions are usually with long tails. For instance, though the median duration of a trip with Vélo'v is 11 minutes and the average is little bit less than 30 minutes, there are rentals lasting more than 2 hours, and the distribution has a tail that is roughly a power-law (Borgnat, Abry, Flandrin, Robardet, Rouquier, and Fleury 2011).

All these studies give a good empirical description of the global features of Vélo'v system along time or space. In the following, we present joint analysis over these two dimensions.

2.2.2 Vélo'v as a complex network

Complex networks are usually employed when studying real-world dataset including some relational properties. In the context of shared bicycle systems, a network arises when looking at stations as vertices of a complex network. Let us define $\mathcal{N}$ the set of stations. Each vertex $n \in \mathcal{N}$ is at a specific geographical place in the city. Going from one station to any other is theoretically possible and around half of all theoretically possible trips have been done at least once. However some preferred trips appear in the data, and they are not necessarily local in space. This justifies the representation of the Vélo'v system as a weighted network.

The relation among the vertices of the Vélo'v network – the stations – is created by the trips made from one station to another: The greater the number of trips made, the more linked the stations are. Let us define $\mathcal{D} = \{(n,m,\tau)\}$ as the set of individual trips going from station $n \in \mathcal{N}$ to station $m \in \mathcal{N}$ at time τ. The Vélo'v network is defined as $\mathcal{G} = (\mathcal{N}, \mathcal{E}, T)$ where the set of possible edges is $\mathcal{E} \subseteq \mathcal{N} \times \mathcal{N}$ and T is a function defining a weighted adjacency matrix varying in time. Let us define $\mathcal{T}$ as a set containing the times t of interest, and $\mathcal{S}$ a set of timescales Δ for aggregation. The function $T : \mathcal{E} \times \mathcal{T} \times \mathcal{S} \to \mathbb{N}$ is obtained by

$$T[n,m](t,\Delta) = |\{(n,m,\tau) \in \mathcal{D} \text{ such that } t \leq \tau < t+\Delta\}| \tag{1}$$

The result $T[n,m](t,\Delta)$ can be seen as the adjacency matrix of a weighted directed graph, that represents a snapshot of the Vélo'v network. On each edge, the weight represents the number of bicycles going from station n to station m between times t and $t+\Delta$. More generally, the

network $\mathcal{G}$ is a dynamical network and issues arise when we need to deal both with its spatial nature (the vertices and the edges) and with its temporal evolution as obtained when varying t or the timescale Δ.

Timescales and aggregation in time of Vélo'v networks.

We take into account the cyclic nature of the network: The same pattern repeats itself each week and we estimate T using periodical averaging. For that, let us decide on a timescale Δ and then define a period $P = p\Delta$, $p \in \mathbb{N}$. Let us set $\mathcal{T}_P = \{k\Delta \mod P; k \in \{0, \dots, p-1\}\}$ so that when applying equation (1), the first interval in which the trips are counted is: $[0 \ \Delta] \mod P$. The periodical estimation of the Vélo'v network is obtained by dividing T by the number of periods P in the data. As the main period in the data is the week (Borgnat, Abry, Flandrin, and Rouquier 2009), we let P be equal to 1 week and Δ=2h for all the results displayed hereafter.

It is possible to reduce the dimension in time of this evolving network by using a Principle Component Analysis in time, as in Borgnat, Abry, Flandrin, Robardet, Rouquier, and Fleury 2011 and Borgnat, Fleury, Robardet, and Scherrer 2009. It turns out that the principal components display peaks in their time evolution that correspond exactly to the different peaks already commented for the global number of rentals, as shown in Figure 2.10. In the following, we will keep in the dynamical adjacency matrix $T[n,m](t,\Delta)$, only the 19 peaks of activity in time, as given by the global behavior as well as the principle components: Every ordinary days around 8am, 12am and 5pm, and each of the two week-end days around 12am and 4pm. We note $\mathcal{T}_P^*$ this set of peak. The Vélo'v network is thus represented by $T[n,m](t,\Delta)$ with $t \in \mathcal{T}_P^*$.

To make possible the comparison of the different snapshots, we propose to build an aggregate of the entire network. Classical aggregation in time is to sum over time the different snapshots and thus focuses on the strong exchanges between stations. In a sense, it can be viewed as going from one timescale to a larger one (hence giving a crude version of multiresolution analysis). Using the 19 peak activity times $\mathcal{T}_P^*$, an aggregated view over the whole week is obtained by $< T[n,m] >_{\mathcal{T}_P^*} = \sum_{t \in \mathcal{T}_P^*} T[n,m](t,\Delta)$.

2.2.3 Aggregation in space for the Vélo'v network

The series of snapshots of graphs convey detailed information, yet this is too much information for modeling. However, aggregating over all the vertices as done in Section 2.2.1 does not give enough details. We would like to aggregate on intermediate spatial scales for the vertices in the network. There are two classical approaches: Find clusters of vertices that are strongly linked together, also called communities (Fortunato 2010), or use multiscale harmonic decomposition over a graph (Hammond, Vandergheynst, and Gribonval 2011). Here, we explore the spatial aggregation that is obtained by looking at communities of stations. So as to compare with the urban organization of the city, a map of the city is in Figure 2.11 that shows the main lines of transportations and provides the places and names of the most important hubs for public transportation. By referring to this map, the reader will follow with greater ease the comments about the spatial aggregation proposed by community aggregation in this section.

Aggregation of network by communities.

At a given timescale and instant, we propose to aggregate the network in space over its communities of vertices. A community is often defined as a subset of vertices that are strongly linked together inside the network. We adopt the modularity as a metric to find communities. Modularity was first proposed in Newman and Girvan 2004 and extended in Leicht and Newman 2008 to the case of directed networks. Assume that t and Δ are set to specific values and use the

Map of Lyon with Velo'v stations: Voronoi
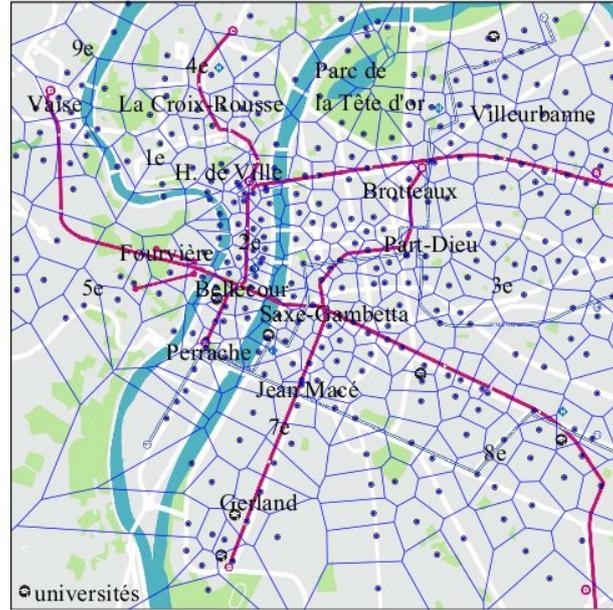
Figure 2.11: Map of the cities of Lyon and Villeurbanne with the Vélo'v stations and their Voronoi diagram (Preparata and Shamos 1985). Each dot is a Vélo'v station surrounded by its own Voronoi cell. One sees that the city is well covered, with a higher density of stations in the center. Major roads are in white; main public transport lines are in red (subways) and grey (tramways). Parks are in green and the two rivers are in blue (the Saône on the west side, the Rhône on the east side). Names of the different parts of the city are given, as well as names of main hubs of transportations: Part-Dieu, Perrache for the main train stations; Vaise and Jean Macé for secondary train stations; Bellecour, Hôtel de Ville, Brotteaux (including Charpennes), Saxe-Gambetta for other important hubs of the public transport system. Gerland, Croix-Rousse and Villeurbanne are other parts of the city that will be discussed afterwards. Finally, the locations of the downtown university campuses are shown.

adjacency matrix $T[n,m](t,\Delta)$ obtained that way. Modularity is defined as:

$$Q = \frac{1}{2W} \sum_{\{n,m\} \in \mathcal{N} \times \mathcal{N}} \left[T[n,m] - \frac{\sum_{j \neq n} T[j,n] \cdot \sum_{k \neq m} T[m,k]}{2W} \right] \delta_{c_n,c_m}$$

where $W = \sum_{n,m} T[n,m]$ is the total weight of the network and c_n is the partition index of the group in which n is. The modularity is equal to the weighted sum of edges inside clusters (as opposed to crossing between clusters), minus the expected weighted sum of such edges if the graph was random conditioned on its degree distribution. Its values range between -1 and $+1$. If there is a community structure in the graph and the index c_n reflects this structure (by taking a different value for each community), Q should be large, typically larger than 0.4. Conversely, if one finds a partition index c_n for which Q is large enough, it tells that there are communities of vertices. As a consequence, finding communities is possible by maximizing Q over the set of $\{c_n, n \in \mathcal{N}\}$ of possible partitions of vertices. However, this task is hard: The complete maximization is NP-complete (Brandes et al. 2008) and many approximations such as the one in Clauset, Newman, and Moore 2004, have a tendency to propose too big communities. In this work, we use the fast, hierarchical and greedy algorithm proposed in Blondel, Guillaume, Lambiotte, and Lefebvre 2008 (called the Louvain algorithm), as a simple way to find relevant communities. It is reviewed in Fortunato 2010 that modularity is a good metric to find communities and that this algorithm works correctly as compared to other methods.

Aggregation in space and time of the Vélo'v network

The method is first applied to the time-aggregated network $< T[n,m] >_{\mathscr{T}_P^*}$. Figure 2.12 shows the community structure obtained by approximate maximization of the modularity Q by the Louvain algorithm. Four communities appear in this network and they are displayed on the Figure. The main feature is that the obtained communities, when shown on a map using the GPS coordinates of each Vélo'v station, are easily grouped on a geographical basis. This can be surprising as the partitioning in communities is blind to any geographical consideration. Anyway, one recognizes in the proposed communities a partition of Lyon city that reflects its general organization. The center of the city is spread out between the Presqu'île (between the two rivers, the Saône on the west, the Rhône on the east) and Part-Dieu (transport hub comprising the main railway station and a subway station) (blue community); the north-east part contains the 6th district and Villeurbanne which are well connected together with a major science university campus in the north (red community); the south and south-east parts are organized along two major roads (one from Gerland to Saxe-Gambetta and then Part-Dieu, a second along the limit between the 3rd and the 8th district to Saxe-Gambetta) (black community on the map). Finally, the north (Croix-Rousse) and north-west (Vaise, 9th district) are separate from other parts because Croix-Rousse is on the top of a high hill, and Vaise accessible only along the Saône river between Croix-Rousse hill and Fourvière hill (5th district); this creates a fourth separate community.
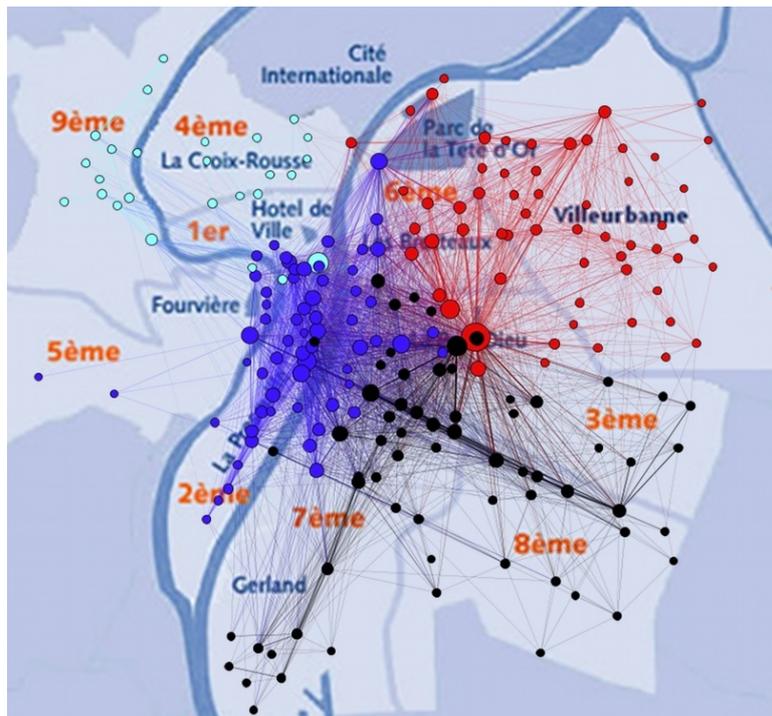
Figure 2.12: Communities of the sum network $< T[n,m] >_{\mathscr{T}_P^*}$: each community has its own color and the size of each vertex is proportional to the number of trips made to and from this station; the width of each edge is proportional to the number of trips made between two stations. For the sake of clarity, the undirected version of the graph is shown and stations with a degree smaller than 2 and edges with less than one trip per week are not shown.

Communities in Vélo'v networks during the week-days.

Communities can also be looked at for individual snapshots $T[n,m](t,\Delta)$, with $t \in \mathscr{T}_P^*$, of the Vélo'v network. Figure 2.13 displays the community structure obtained for two snapshots taken

during ordinary job days (here, on Mondays). The community structure at a given time in the week still matches a geographical partitioning of the city. Because the timescale is finer, with less aggregation in time, more details are apparent and some specific Vélo'v stations are not in the same community as their surrounding stations. For instance, the northern community (Villeurbanne) which includes a major university campus, includes also a station on the banks of the Rhône where there is also a university. Also, there are more communities (6 here) than in the average network. One new community groups together the Croix-Rousse hill with the Hôtel de Ville and the 6th district which contain the closest downhill subway stations. The comparison of the maps in the morning (on the left) and at the end of the afternoon (on the right) is interesting: It shows that most of the communities are left unchanged. This indicates that Vélo'v, like other ordinary transportation means, is used for commuting (from home to work in the morning, and back in the late afternoon). However, the community grouping the Croix-Rousse hill and the 6th district is not present anymore: It is not hard to figure out that people living on the Croix-Rousse will not use Vélo'v bicycles (which are heavy bicycles) to go up the hill back to their home. Apart from that, more than 90% of the remaining vertices did not change of community. Note that the same results are obtained for other ordinary week-days.
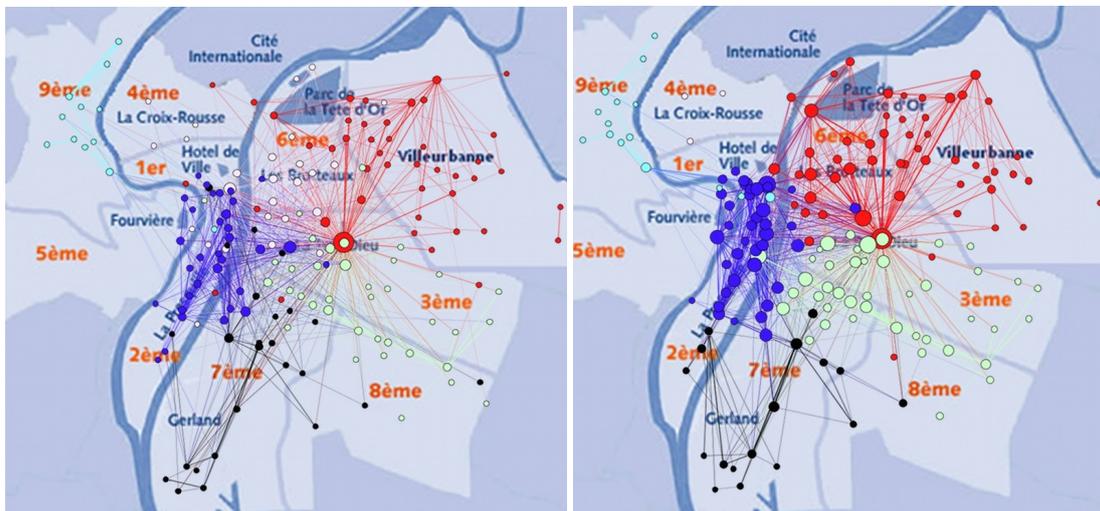
Figure 2.13: Aggregation in space of snapshots of the Vélo'v network: Monday 7am-9am (left) and Monday 4pm-6pm (right). Each community has an arbitrary color. The size of each vertex is proportional to its incoming flow added to its outgoing flow. Note that the number of trips is usually larger during the afternoon (as was already seen on Figure 2.10, the peak for two hours in the afternoon is 1.5 times higher than the one in the morning).

Communities in Vélo'v networks during the week-ends.

When turning to the analysis of the week-end uses of Vélo'v, the features change a little bit as shown in Figure 2.14. A first point is that the most active stations during the week-ends are not always the same than during the ordinary days. Major transportation hubs are unchanged (Part-Dieu, down-hill of Fourvière, Hôtel de Ville,...) yet new active stations appear near places for shopping (in the Presqu'île) and all around the large and green city park of la Tête d'Or in the north.

On Sunday (on the right), the communities could be reminiscent of the time-averaged ones excepted on two points. First, Vaise (9th district) is grouped with the Presqu'île community, possibly because the paths along the river are a pleasant leisure trip. Second, the community ranging from Part-Dieu to the 3rd and 8th districts contains a station which is the most active during the week-ends: The station at the entrance of the city park of la Tête d'Or. Also some
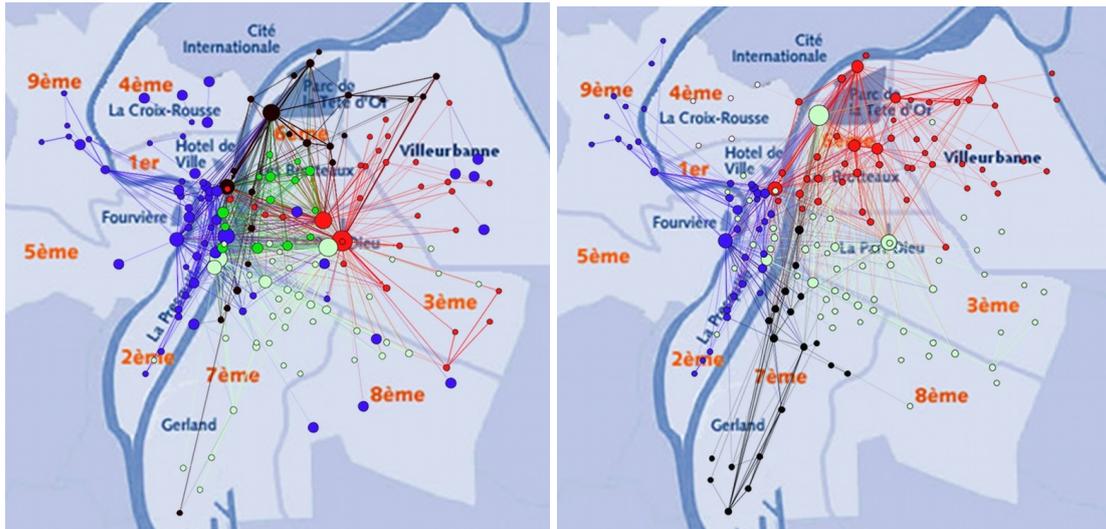
Figure 2.14: Aggregation in space of snapshots of the Vélo'v network Saturday 3pm-5pm (left) and Sunday 3pm-5pm (right). Each community has an arbitrary color.

stations along the Rhône river are grouped in the same community. Here again, this is not a great surprise as the park of la Tête d'Or is a main destination for Sunday's outdoor activities. This community connects this park to other places that are either hubs of transportation (like Part-Dieu and Saxe-Gambetta) or other subway stations.

For Saturdays (on the left), the situation is more complex and does not reflect easily a simple geographical partition of the city. A community organized around the park of la Tête d'Or and grouping many stations around the park and on the river banks (having an easy access to the park thanks to bicycle paths on the river banks) is clearly visible. A surprising feature is that the periphery (Vaise, Croix-Rousse but also the most eastern parts of the city) is grouped in the community of the city center (in blue). This is a clue showing that some people uses Vélo'v for longer distance trips on Saturdays than on ordinary days.

This last aspect is one example of the fine scale analysis that are made possible by aggregating the network in a meaningful manner. It helps finding some unexpected structure that could be probed with more details in the complete dataset.

2.2.4   Typology of dynamics of the Vélo'v network

Another method for aggregation of vertices is possible: We now want to group two vertices if they have the same usage pattern, whereas in the previous section we grouped vertices exchanging many bicycles. Such an aggregated view is different from summing up the individual snapshots. With the objective of proposing a streamlined methodology for aggregating networks in space and/or time, we will show how the notion of communities can be tailored to group vertices with similar behaviors. For that, the idea is to first build a new similarity network from the dynamical network, before finding communities of similar vertices.

Similarity graph for the dynamics.

The principle is to quantify the resemblance over time of the different flows between stations. For that, one considers each snapshot to be one observation of the network, and then builds a similarity matrix between stations based on these observations. Given a station $n \in \mathcal{N}$, two feature vectors characterize its activity: The incoming flows $F^{in}[n](t) = \sum\limits_{i \in \mathcal{N}} T[i,n](t,\Delta)$ and the outgoing flows $F^{out}[n](t) = \sum\limits_{j \in \mathcal{N}} T[n,j](t,\Delta)$ where $t \in \mathcal{T}_P$ and Δ is constant. For a given

pair of stations $[n,m]$, quantities can be computed to quantify if these activity patterns look alike or not. A general approach relies on the choice of a distance d between features (see for instance Basseville 1989 for many possible distances, or Fortunato 2010 for application on graphs), leading to distances between activities of stations n and m:

$$D^{in}[n,m] = d(F^{in}[n], F^{in}[m]) \quad \text{and} \quad D^{out}[n,m] = d(F^{out}[n], F^{out}[m]).$$

For dealing with observations at different times $t \in \mathscr{T}_P$, it is natural to use a correlation distance over the various observations. The empirical estimator of correlations reads as

$$D^{in}_{\mathscr{T}_P}[n,m] = \frac{1}{|\mathscr{T}_P|} \sum_{t \in \mathscr{T}_P} \tilde{F}^{in}[n](t) \tilde{F}^{in}[m](t)$$

where $\tilde{F}^{in}[n]$ is the centered and normalized version for each n of $F^{in}[n]$ (respectively for $\tilde{F}^{out}[n]$).

When looking at individual snapshots of the Vélo'v network, we have commented that the behaviors during the week-days are roughly unchanged from one day to another. It makes sense to take as the set of relevant times $\mathscr{T}_P$ the 15 peaks of activity of the week-days that were already used previously: 8am, 12am and 5pm, with $\Delta = 2$h. We obtain two correlation matrices of size $|\mathscr{N}| \times |\mathscr{N}|$, that we note D^{in}_{week} and D^{out}_{week}. For the week-ends, the behavior of the stations is different and we compute separately a correlation for the features during the week-ends. Using for $\mathscr{T}_P$ the times 12am and 4pm of Saturday and Sunday and a timescale $\Delta = 2$h, two other correlation matrices are obtained: $D^{in}_{\text{w.-end}}$ and $D^{out}_{\text{w.-end}}$.

Remind that the goal is to compare the behaviors of stations, hence the choice of looking at in-in or out-out correlations. An alternative would be to study in-out correlations between pairs of stations; this metric would describe whether two stations are well connected in the meaning that bicycles leaving one station have a good chance to arrive at another station. However, this metric appear to be less interesting: First, the mere study of the flows connecting two stations, as studied in Section 2.2.3, gives already a picture of how well two stations are connected in this acceptance and this new study would be somewhat redundant; second, the flows leaving a given station are usually really spread between many other stations: The statistical confidence on estimated in-out correlations is low.
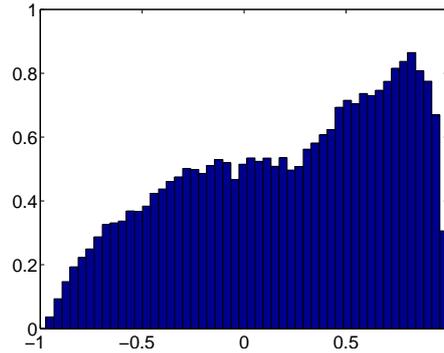
Figure 2.15: Distribution of the empirical correlations D^{out}_{week} values.

For the Vélo'v network, the situation is that of many vertices ($|\mathscr{N}| = 338$) but only a few observations on one dataset because $|\mathscr{T}_P|$ is 15 for week-days, and 4 for week-ends. Recent theoretical studies about *Correlation Screening* (Hero and Rajaratnam 2011) have shown that, even under the null hypothesis of no correlations between the vertex features, one should expect large estimated values if using the empirical correlation for large $|\mathscr{N}|$ and small $|\mathscr{T}_P|$. The number of false discovery of non-zero correlations can then be really large. In Hero and

Rajaratnam 2011, expressions are given to estimate the threshold under which false discovery becomes dominant. As a consequence, if one wants to build a network of similarity between vertices based on correlation for a small number of observations (as it is often the case), it is expected that a thresholding operation is needed on the correlation to reduce the number of false discoveries. Using Hero and Rajaratnam 2011, and the specific values for the Vélo'v network, a threshold in correlation of 0.8 is reasonable to obtain some statistical confidence of discovering real correlations. Figure 2.15 shows the histogram of the values of the correlation matrix D^{out}_{week} outside the diagonal. A maximum in the probability of finding large correlations occurs around 0.85 that is not predicted by a null hypothesis of uncorrelated vertex features. This is a sign of existing similarities in the vertices' activities in the network.

We build a similarity graph for the vertices by thresholding the 4 correlation matrices and summing up the thresholded correlations. The weight $S[n,m]$ on each edge of the similarity graph is then

$$S[n,m] = \sum_{\substack{dir.=\{in,out\} \\ time=\{week,w.-end\}}} \delta_{(D^{dir.}_{time}[n,m]>\eta)} \dot{D}^{dir.}_{time}[n,m]$$

where η is the threshold. Based on the analysis before, we set $\eta = 0.8$ (though the results are not sensitive to small changes of η). An important remark is that the threshold is applied directly on D, not on its absolute value: For the Vélo'v network, negative correlations are discarded because stations with opposite activities would not be in a group of similar behavior.

As a consequence, the similarity weights $S[n,m]$ are between 0 and 4. If two vertices are never similar, neither during the week nor during the week-ends, both for incoming and leaving flows, the weight is 0 and the vertices (i.e., Vélo'v stations) are not connected in the similarity graph. If the vertices have similar behavior along time for some of the features, the weight will increase by being higher than η, 2η, 3η or 4η if they are similar for one, two, three or all of the feature correlation matrices. Using thresholding before summing the correlations allows us to escape the poor estimation in correlation screening.

Communities of dynamical activities.

Given the similarity graph, quantifying if the activities of two stations look alike along time or not, it is possible to build a typology of the stations by grouping them according to these correlations. This is simply framed as a problem of detecting communities in the similarity graph of weights S. The same method of community detection is used on the weighted similarity graph. It provides communities of stations that share their pattern of activity in time. Each community is a type of dynamical activity in the Vélo'v network. The set of communities can be seen as a typology of the different dynamics at work in the network. Figure 2.16 shows the obtained typology on the Lyon map.

As compared to the previous communities obtained for space aggregation, the similarity communities can not be matched on a simple geographical partitioning of the city. However, it can be interpreted as a kind of segmentation of the city in various zones of activity. For instance, the community in black groups most of the vertices from the university campus (Villeurbanne and near the park in the north, Gerland in the south, the medicine university in the east and the university on the banks of the Rhône) and parts of the city with many companies – places to which people commute. The community of Part-Dieu (in red) includes many places of major subway or tramway hubs. Another (in dark blue) is spread out in the city center and has extensions along the stations of the subway lines crossing the center at Bellecour. Finally, the two remaining communities (in light colors) group parts of the cities that are in the east (mostly residential area) or near the Saône river banks in the center (where there are many shops, especially active during the week-ends).
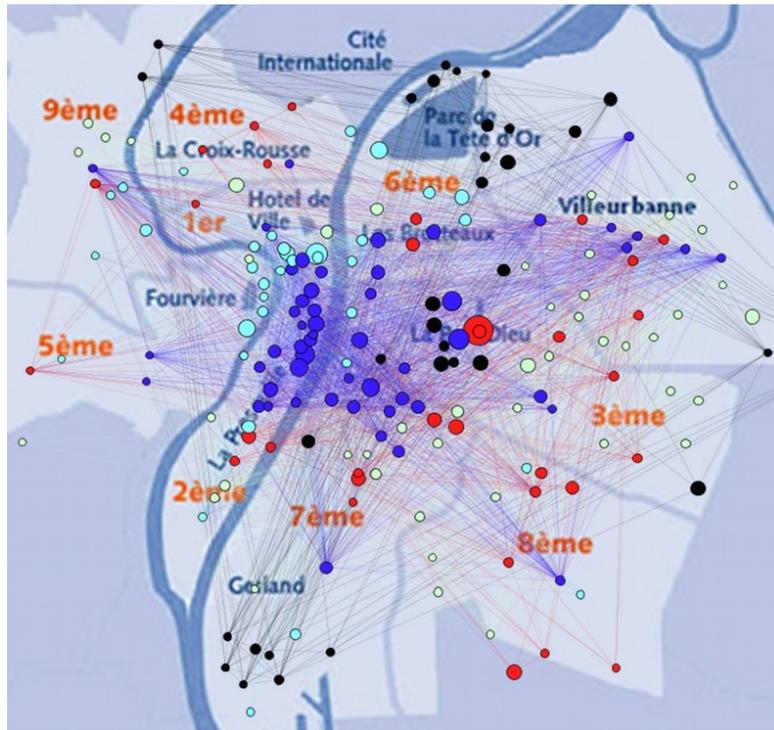
Figure 2.16: Communities in the similarity graph $S[n,m]$ of the Vélo'v network. Each community (in an arbitrary color) reflects a specific pattern of usage of the station during the week. Edges are non-zero weights of similarity $S[n,m]$ and vertices have a size proportional to the number of trips made to and from this station.

2.3 Discussion

Statistical time-series analysis and structural techniques can be used to reveal the main global properties of relational dynamic graphs. From the sensor mobility networks studied, it appears that all graph properties are highly correlated, except for the links creation and deletion processes which are independent of other graph properties. The dynamics of structures, such as the triangles or the connected components, are key features of the evolution of relational graphs: The dynamics of triangle creation is higher than what would be expected in non-structured random graphs and the connected components are less stable than what was expected. Based on these analysis, simple yet very accurate models can be designed that generate random relational dynamic graphs. The qualitative summaries of the trajectories of vertices among few identified structures of the graph are very similar, whatever they are computed on the original data or on data generated by the models.

Community detection mining tools can also be used to extract clusters of vertices based on an intra versus inter community interaction measure (modularity). In the analysis of the Vélo'v dataset, it extracts either communities of stations that exchange regularly a large number of bicycles, or communities of stations that are similar in the time patterns of bicycle flows. Such analysis enabled us to gain a significant understanding on the social usage of the Vélo'v program in Lyon: Communities remain geographically concentrated (hence indicating a preferred short-range use of the bicycles) while time patterns of flows between stations display similarities so that they are grouped in clusters separating trips related to professional activities (week days and major communication hubs) from those used during leisure time (week end and parks).

The analysis of Vélo'v data is still an on going work that is conducted within Vél'innov

ANR project [2]. Two directions of research are currently under study: (1) we are considering the extraction of relationships between Vélo'v station attribute changes (the characteristics of the population living or working near by, the place of interest in their neighborhood) and the topological modifications of the graph, that is to say the flow between stations; (2) we are considering the use of signal processing methods to the study of these data by transforming the graph into signals, and analyzing them using signal processing methods (Hamon, Borgnat, Flandrin, and Robardet 2013a; Hamon, Borgnat, Flandrin, and Robardet 2013b).

---

[2]Projet ANR Sociétés Innovantes, innovation, économie, modes de vie – INOV 2012

3 — Mining attributed dynamic graphs

Whereas in the previous chapter we considered modeling and description methods, that aim at understanding the generative processes of the data, we focus here on local pattern discovery in attributed dynamic relational graphs. Such techniques rely on the fast counting of the exhaustive collections of patterns that provide a fairly complete picture of the information content of the graph. Each pattern describes a part of the relational graph that is significant as evaluated by a measure of interest. One of the challenges is to design patterns that describe both the similarity of the vertices relative to the structure encoded by the graph relationship and the similarity of the vertices with respect to their attribute values. As the size of the search space is very large, to be efficient, the extraction must wisely exploit the constraints. In this chapter, we present our work related to constraint-based pattern mining in relational graphs that are either attributed or dynamic, or both. In Section 3.1, we extract local patterns in static attributed graphs that combine information about the connectivity of the vertices and their attribute values (Prado, Plantevit, Robardet, and Boulicaut 2013; Salotti, Plantevit, Robardet, and Boulicaut 2012). The connectivity of each vertex is described by topological properties that quantify its topological status in the graph (connectivity, centrality, etc.). Co-variations among these properties and the numerical attributes associated to each vertex are extracted: Sets of attributes, that share the same tendency among a large number of vertex pairs, describe important local aspect of the relational graph. In Section 3.2, we study dynamic relational graphs by extracting patterns that identify the dynamics of highly connected sub-graphs (Robardet 2009). This approach relies on the local-to-global framework: It first extracts isolated pseudo-cliques and second model the dynamics of the graph as the evolution in time of these patterns. Finally, in Section 3.3, attributed dynamic graphs are considered. The constraint-based pattern mining framework is used to identify dynamic sub-graphs whose vertices follow the same trend over time for a subset of attributes with numerical values (Desmier, Plantevit, Robardet, and Boulicaut 2013). This pattern domain relies on the graph structure and the temporal evolution of the attribute values. Several interestingness measures are proposed to focus on the most relevant patterns with regard to the graph structure, the vertex attributes, and the time.

3.1 Mining attributed relational graphs

Existing methods that support the discovery of local patterns in graphs mainly focus on the topological structure of the patterns, by extracting specific sub-graphs while ignoring the vertex

properties (e.g. cliques (Makino and Uno 2004) or quasi-cliques (Liu and Wong 2008; Uno 2010)), or compute frequent relationships between vertex attribute values (frequent sub-graphs in a collection of graphs (Jiang and Pei 2009) or in a single graph (Bringmann and Nijssen 2008)), while ignoring the topological status of the vertices within the whole graph, e.g., the vertex connectivity or centrality. The same limitation holds for the methods proposed in (Khan, Yan, and K. Wu 2010; Mougel, Rigotti, and Gandrillon 2012; Silva, Meira, and Zaki 2010; Silva, Meira, and Zaki 2012), which identify sets of vertices that share local attributes and that are close neighbors. Such approaches only focus on a local neighborhood of the vertices and do not consider the connectivity of the vertex in the whole graph. To describe static attributed graphs, we propose in Prado, Plantevit, Robardet, and Boulicaut 2013 a broader setting that makes possible to find similarities among vertex descriptors, that is to say vertex attributes all together with topological vertex properties. These descriptors are mostly of numerical or ordinal types, and we propose to capture their similarity by quantifying their co-variation to indicate how vertex descriptors of a set tend to monotonically increase or decrease all together.

Let us illustrate our proposal on a co-authorship graph depicted in Figure 3.1, where vertices (from A to P) denote authors, edges encode co-authorship relations, and three attributes describe author: h corresponds to the author h-index, which attempts to measure both the productivity and the impact of the published work of each author (Hirsch 2005); i denotes the average number of hours per week spent by each author on instructional duties; and t designates the number of publications the author had in the IEEE TKDE journal. As topological property, we consider the betweenness centrality measure, that is the number of times a vertex appears on a shortest path of the graph (see Section 3.1.1). This value is shown in a circle associated to each vertex. For instance, vertex D has attribute values $h = 25$, $i = 1.5$ and $t = 18$ and a betweenness centrality value equal to 73. One of the topological patterns extracted from this attributed graph is $P = \{h^+, i^-, \text{BETW}^+\}$, whose meaning is *the higher the value of attribute h, the lower the value of attribute i and the higher the betweenness centrality of a vertex*. In other words, authors that tend to have a high h-index, tend to have a low instructional duty and publish articles with co-authors that are also central in the graph, inducing a rather small distance to other vertices. This topological pattern combines a topological property (BETW) with two vertex attributes (h and i) and is supported by 89 pairs of vertices among the $\binom{16}{2}$ possible pairs over the graph. Its top 3 representative vertices are E, I and D (shadowed on Figure 3.1). These vertices have the highest values on h and BETW, and the lowest values on attribute i compared to other vertices. Therefore, these dominant vertices have a significant impact on the support of this pattern.

3.1.1 Topological pattern domain

The input of our mining task is a non-directed attributed graph $G = (V, E, L)$, where V is a set of n vertices, E a set of m edges, and $L = \{l_1, \cdots, l_p\}$ a set of p attributes associated with each vertex of V, which may be numerical or ordinal. Important properties of the vertices are also encoded by the edges of the graph, which describe inter-relations between vertices. From this relation, we can compute some topological properties that synthesize the role played by each vertex in the graph. The topological properties we are interested in range from a microscopic level – those that describe a vertex based on its direct neighborhood – to a macroscopic level – those that characterize a vertex by considering its relationship to all other vertices in the graph. Statistical distributions of these properties are generally used to depict large graphs (see, e.g., (Albert and Barabási 2000; Kang, Tsourakakis, Appel, Faloutsos, and Leskovec 2011)).

Microscopic properties

We propose to use four topological properties to describe the direct neighborhood of a vertex v:

- The degree of v is the number of edges incident to v ($deg(v) = |\{u \in V, \{u, v\} \in E\}|$).

Figure 3.1: A co-authorship attributed graph toy example.

When normalized, it is called the degree centrality coefficient: $\text{DEGREE}(v) = \frac{deg(v)}{n-1}$ (e.g., $\text{DEGREE}(B) = \frac{3}{15}$).

- The clustering coefficient evaluates the connectivity of the neighbors of v and thus its local density:

$$\text{CLUST}(v) = \frac{2|\{\{u,w\} \in E, \{u,v\} \in E \wedge \{v,w\} \in E\}|}{deg(v)(deg(v)-1)}$$

(e.g., $\text{CLUST}(B) = \frac{2|\{\{A,D\}\{C,D\}\}|}{3 \times 2} = \frac{2}{3}$)

- To better understand the structure of the neighborhood of v, we also consider the quasi-cliques (Liu and Wong 2008) that involve v. v belongs to a γ-quasi clique Q iff the graph G_Q induced by the set of vertices Q is connected and satisfies

$$\forall u \in Q, deg_{G_Q}(u) \geq \lceil \gamma(|Q|-1) \rceil$$

where $deg_{G_Q}(u)$ is the degree of u in G_Q (e.g., $\{A,B,C,D\}$ is a 2/3-quasi clique since $deg_{G_Q}(A) = deg_{G_Q}(c) = 2 \geq \lceil \frac{2}{3}(4-1) \rceil$ and $deg_{G_Q}(B) = deg_{G_Q}(D) = 3 \geq 2$). We consider two properties based on the quasi-cliques involving v: the size of the largest quasi-clique ($\text{SzQC}(v)$) and the number of quasi-cliques ($\text{NBQC}(v)$).

Macroscopic properties

We consider five macroscopic topological properties to characterize a vertex while taking into account its connectivity to all other vertices of the graph.

- Vertex communities can be computed by looking for a partition of V that maximizes the Newman's modularity measure (Newman 2004). This criterion is based on the proportion of edges that fall within the community minus the expected such proportion if edges were distributed at random:

$$Q = \frac{1}{4m} \sum_{u,v} \left(\mathbb{1}_E(\{u,v\}) - \frac{deg(u)deg(v)}{2m} \right) \delta_{c_u,c_v}$$

where c_v is the community assigned to v, δ_{c_u,c_v} is the Kronecker delta ($\delta_{c_u,c_v} = 1$ if $c_u = c_v$ and $\delta_{c_u,c_v} = 0$ otherwise), $\mathbb{1}_E(\{u,v\})$ is the indicator function of the set E ($\mathbb{1}_E(\{u,v\}) = 1$ if $\{u,v\} \in E$, 0 otherwise). For example, according to such a definition, the 4 communities on Figure 3.1 are $\{A,B,C,D\}$, $\{E,N,O,P\}$, $\{F,G,H\}$, $\{I,J,K,L,M\}$. As topological property, we consider the size of the community of v ($\text{SzCOM}(v)$).

- The relative importance of vertices in a graph can be obtained through centrality measures (Freeman 1977). Closeness centrality $\text{CLOSE}(v)$ is defined as the inverse of the average distance between v and all other vertices that are reachable from it. The distance between two vertices is defined as the number of edges of the shortest path between them: $\text{CLOSE}(v) = \frac{n}{\sum_{u \in V} |shortest\_path(u,v)|}$ (e.g., $\text{CLOSE}(B) = 0.021$ and $\text{CLOSE}(E) = 0.037$).
- The betweenness centrality $\text{BETW}(v)$ of v is equal to the number of times a vertex appears on a shortest path in the graph. It is evaluated by first computing all the shortest paths between every pair of vertices, and then counting the number of times a vertex appears on these paths: $\text{BETW}(v) = \sum_{u,w} \mathbb{1}_{shortest\_path(u,w)}(v)$ (e.g., $\text{BETW}(B) = 1$ and $\text{BETW}(E) = 166$).
- The eigenvector centrality measure (EGVECT) favours vertices that are connected to vertices with high eigenvector centrality. This recursive definition can be expressed by the following eigenvector equation $Ax = \lambda x$ which is solved by the eigenvector x associated to the largest eigenvalue λ of the adjacency matrix A of the graph (e.g., $\text{EGVECT}(B) = 0.093$ and $\text{EGVECT}(E) = 0.114$).
- The PAGERANK index (Brin and Page 1998) is based on a random walk on the vertices of the graph, where the probability to go from one vertex to another is modelled as a Markov chain in which the states are vertices and the transition probabilities are computed based on the edges of the graph. This index reflects the probability that the random walk ends at the vertex itself: $\text{PAGERANK}(v) = \alpha \sum_j \mathbb{1}_E(\{u,v\}) \frac{\text{PAGERANK}(u)}{deg(u)} + \frac{1-\alpha}{n}$, where the parameter α is the probability that a random jump to vertex v occurs (e.g., $\text{PAGERANK}(B) = 1.11$ and $\text{PAGERANK}(E) = 1.50$).

These 9 topological properties characterizes the graph relationship encoded by E. These properties, along with the set of vertex attributes L, constitutes the set of numerical attributes $\mathcal{M}$. The numerical dataset $\mathcal{D}$ to be mined associates to each vertex $v \in V$ a numerical value on each attribute of $m \in \mathcal{M}$: $m(v) = w$ means that w is the value of attribute m for vertex v.

Topological patterns

Topological patterns are sets of numerical attributes that behave similarly over a large part of the vertices of the graph. The co-variation of an attribute with respect to others is denoted by a sign $\{+,-\}$: $+$ if the attribute co-varies the same way as the other attributes, $-$ if it co-varies the opposite way. Thus, considering signed attributes $\mathcal{M} \times \{+,-\}$, the language of patterns is defined as $\mathcal{L} = 2^{\mathcal{M} \times \{+,-\}}$. For convenience, the signed attribute $(m,s) \in \mathcal{M} \times \{+,-\}$ is denoted m^s. Following the example of Figure 3.1, the trend "*the more papers in IEEE TKDE (t) the lower the average number of hours per week spent on instructional duties (i)*" is represented by the pattern $\{t^+, i^-\}$.

Definition of constraints

Several signed vertex descriptors co-vary if the orders induced by each of them on the set of vertices are consistent. This consistency is evaluated by the vertex pairs ordered the same way by all descriptors. The number of such pairs constitutes the so-called support of the pattern. This measure can be seen as a generalization of the Kendall's τ measure (Kendall 1948). When we consider all possible vertex pairs, this interestingness measure is defined as follows:

> **Definition 3.1** — $Supp_{all}$.   The support of a topological pattern $P \in \mathcal{L}$ over all possible pairs of vertices is:
> $$Supp_{all}(P) = \frac{|\{(u,v) \in V^2 \mid \forall m^s \in P : m(u) \rhd_s m(v)\}|}{\binom{n}{2}}$$
> where $\rhd_s$ denotes $<$ when s is equal to $+$, and $\rhd_s$ denotes $>$ when s is equal to $-$.

This measure gives the number of vertex pairs (u, v) such that u is strictly lower than v on all descriptors with sign $+$, and u is strictly higher than v on descriptors with sign $-$. For instance, the pattern $P = \{t^+, i^-\}$ is supported by 85 pairs among the 120 possibles ones, hence $Supp_{all}(P) = 0.71$.

Emerging pattern constraints: To identify most interesting topological patterns, we propose to give to the end-user the possibility of guiding its data mining process by querying the patterns with respect to their correlation with the relationship encoded by the graph or with a selected descriptor. Therefore, we revisit the notion of emerging patterns (Dong and Li 1999) by identifying the patterns whose support is significantly greater (i.e., according to a growth-rate threshold) in a specific subset of vertex pairs than in the remaining ones. This subset can be either defined as the vertex pairs that are ordered with respect to a selected descriptor called the class descriptor, or it can be E, the set of edges of the graph. Whereas the former highlights the correlation of a pattern with the class descriptor, the latter enables to characterize the importance of the graph structure within the support of the topological pattern. For instance, considering the toy example of Figure 3.1, h^+t^+ and h^+t^- are both frequent with minimum support of 20%. Note that although these patterns are contradicting, they are both output by our approach when only the frequency constraint is considered. The extraction of emerging patterns with respect to t outputs the pattern h^+t^+ as the frequency of h^+ is significantly greater in t^+ than in t^- (with a factor of 2.13). h^+t^+ is more emerging with respect to E than h^+t^-, their growth rates being respectively equal to 1.23 and 0.59.

Let us consider a selected attribute $A \in \mathcal{M}$ and a sign $r \in \{+, -\}$. The set of pairs of vertices that are ordered by A^r is $\mathscr{P}_{A^r} = \{(u, v) \in V^2 \mid A(u) \rhd_r A(v)\}$. The support measure based on the vertex pairs of $\mathscr{P}_{A^r}$ is defined below.

> **Definition 3.2** — $Supp_{A^r}$. The support of a topological pattern P over A^r is:
>
> $$Supp_{A^r}(P) = \frac{|\{(u, v) \in \mathscr{P}_{A^r} \mid \forall m^s \in P : m(u) \rhd_s m(v)\}|}{|\mathscr{P}_{A^r}|}$$

Analogously, the support of P over the pairs of vertices that do not belong to $\mathscr{P}_{A^r}$ is denoted $Supp_{A^{\bar{r}}}(P)$. To evaluate the impact of A^r on the support of P, we consider the growth rate of the support of P over the partition of vertex pairs $\{\mathscr{P}_{A^r}, \mathscr{P}_{A^{\bar{r}}}\}$: $Gr(P, A^r) = \frac{Supp_{A^r}(P)}{Supp_{A^{\bar{r}}}(P)}$

If $Gr(P, A^r)$ is greater than a minimum growth-rate threshold, then P is referred to as emerging with respect to A^r. If $Gr(P, A^r) \approx 1$, P is as frequent in $\mathscr{P}_{A^r}$ as in $\mathscr{P}_{A^{\bar{r}}}$. If $gr(P, A^r) \gg 1$, P is much more frequent in $\mathscr{P}_{A^r}$ than in $\mathscr{P}_{A^{\bar{r}}}$. For example, $Gr(\{h^+, i^-, \text{BETW}^+\}, t^+) = 2.31$. The intuition behind this definition is to identify the topological patterns that are mostly supported by pairs of vertices that are also ordered by the selected descriptor.

Emerging patterns w.r.t. the graph structure: It is interesting to measure if the graph structure plays an important role in the support of a topological pattern P. To this end, we define a similar support measure based on pairs that belongs to E, the set of edges of the graph:

$$\mathscr{P}_E = \{(u, v) \in V^2 \mid \{u, v\} \in E\}$$

Based on this set of pairs, we define the support of P as:

> **Definition 3.3** — $Supp_E$. The support of a topological pattern P over the pairs of vertices

that are linked in G is:

$$Supp_E(P) = \frac{2|\{(u,v) \in \mathscr{P}_E \mid \forall m^s \in P : m(u) \rhd_s m(v)\}|}{|\mathscr{P}_E|}$$

The maximum value of the numerator is $\frac{|\mathscr{P}_E|}{2}$ since: (1) if $(u,v) \in \mathscr{P}_E$ then $(v,u) \in \mathscr{P}_E$, and (2) it is not possible that $\forall m^s \in P$, $m(u) \rhd_s m(v)$ and $m(v) \rhd_s m(u)$ at the same time. For instance, the pattern $\{h^+, i^-\}$ is supported by all the twenty possible pairs that are edges, its support is thus equal to 1. The support of P over the pairs of vertices that do not belong to $\mathscr{P}_E$ is denoted $Supp_{\overline{E}}(P)$.

As before, to evaluate the impact of E on the support of P, we consider the growth rate of the support of P over the partition of vertex pairs $\{\mathscr{P}_E, \mathscr{P}_{\overline{E}}\}$:

$$Gr(P,E) = \frac{Supp_E(P)}{Supp_{\overline{E}}(P)}$$

$Gr(P,E)$ enables to assess the impact of the graph structure on the pattern. Therefore, if $Gr(P,E) \gg 1$, P is said to be *structurally* correlated. If $Gr(P,E) \ll 1$, the graph structure tends to inhibit the support of P. For example, on Figure 3.1, the most structurally correlated pattern is $P = \{h^+, t^+, \textsc{Betw}^+\}$ with $Gr(P,E) = 1.628$.

Top k representative vertices: The user may be interested in identifying the vertices that are the most representative of a given topological pattern, thus enabling the projection of the patterns back into the graph. For example, the representative vertices of the pattern $\{t^+, \textsc{Betw}^-\}$ would be researchers with a relatively large number of IEEE TKDE papers and a low betweenness centrality measure.

We denote by $S(P)$ the set of vertex pairs (u,v) that constitutes the support of a topological pattern P: $S(P) = \{(u,v) \in V^2 \mid \forall m^s \in P : m(u) \rhd_s m(v)\}$ It forms with V a directed graph $G_P = (V, S(P))$. This graph satisfies the following property.

Property 3.1   The graph $G_P = (V, S(P))$ is transitive and acyclic.

Proof. Let us consider $(u,v) \in V^2$ and $(v,w) \in V^2$ such that, $\forall m^s \in P : m(u) \rhd_s m(v)$ and $m(v) \rhd_s m(w)$. Thus, $m(u) \rhd_s m(w)$ and $(u,w) \in S(P)$. Therefore, G_P is transitive.

As $\rhd_s \in \{<, >\}$, it stands for a strict inequality. Thus, if $(u,v) \in S(P)$, $(v,u) \notin S(P)$. Furthermore, as G_P is transitive, if there exists a path between u and v, there is also an arc $(u,v) \in S(P)$. Therefore, $(v,u) \notin S(P)$ and we can conclude that G_P is acyclic. ∎

As G_P is acyclic, it admits a topological ordering of its vertices, which is, in the general case, not unique. The top k representative vertices of a topological pattern P are identified on the basis of such a topological ordering of V and are the k last vertices with respect to this ordering. Considering that an arc $(u,v) \in S(P)$ is such that v dominates u on P, this vertex set contains the most dominant vertices on P. The top k representative vertices of P can be easily identified by ordering the vertices by their incoming degree as shown in Section 3.1.2.

Constraint properties

An anti-monotone constraint: As mentioned in (Calders, Goethals, and Jaroszewicz 2006), $\mathscr{C}_{Supp_{all}} \equiv Supp_{all} \geq minsup$, where $minsup$ is a user-defined minimum support threshold, is an anti-monotone constraint for positively signed descriptors. This is still true when considering negatively signed ones: adding m^- to a pattern P leads to a support lower than or equal to that of P since, to contribute to the support, a pair (u,v) that supports P must also satisfy $m(u) > m(v)$.

Besides, when adding descriptors with negative sign, the support of some patterns can be deduced from others, the latter referred to as symmetrical patterns.

> **Property 3.2 — Support of symmetrical patterns.** Let P be a topological pattern and $\overline{P}$ be its symmetrical, that is, $\forall m^s \in P$, $m^{\overline{s}} \in \overline{P}$, with $\overline{s} = \{+,-\} \setminus \{s\}$. If a pair (u,v) of V^2 contributes to the support of P, then the pair (v,u) contributes to the support of $\overline{P}$. Thus, we have $Supp_{all}(P) = Supp_{all}(\overline{P})$.

Topological patterns and their symmetrical patterns are semantically equivalent. Therefore, we avoid the irrelevant computation of duplicate topological patterns by exploiting this property.

An upper Bound on the Support Measure: While it is generally sufficient to use an anti-monotone constraint to make possible the extraction patterns, in this context we face a particular challenge: the support counting is quadratic in the number of vertices. Thus, it is important to avoid, in linear time, some useless support computation. To this end, we derive an upper bound on the support used to safely prune non-promising topological patterns.

To define an upper bound on the support of P which benefits from the presence of ties in the descriptors, a rank value $\rho(m(u))$ is associated with each numerical descriptor value $m(u)$ (Calders, Goethals, and Jaroszewicz 2006). $\rho(m(u))$ is the index of u in V when V is sorted in ascending order with respect to m, such that $1 \leq \rho(m(u)) \leq |V|$, ties being handled arbitrarily. Actually, due to the presence of ties, there are many possible rankings, but in all of them, the ranks of a given value range in an interval defined by $[\underline{\rho}(m(u)), \overline{\rho}(m(u))]$ with:

$$
\begin{aligned}
\underline{\rho}(m(u)) &= \min\{\rho(m(v)) \mid v \in V \text{ and } m(v) = m(u)\} \\
\overline{\rho}(m(u)) &= \max\{\rho(m(v)) \mid v \in V \text{ and } m(v) = m(u)\}
\end{aligned}
$$

For instance, on graph of Figure 3.1, $\underline{\rho}(\text{BETW}(B)) = 8$ and $\overline{\rho}(\text{BETW}(B)) = 9$. Given two descriptors A and B and their respective signs s_a and s_b, the ranking intervals over these descriptors can be used to establish a lower bound on the number of vertices that cannot form a supporting pair with u. If v_a is a vertex such that $(A(v_a) \unrhd_{s_a} A(u))$, then the pair (u, v_a) cannot support $A^{s_a} B^{s_b}$. On the other hand, if a vertex v_b does not satisfy $(B(v_b) \rhd_{s_b} B(u))$, then the pair (v_b, u) cannot support $A^{s_a} B^{s_b}$ either. We denote I^{s_a} and J^{s_b} the sets of vertices v_a and v_b, respectively. Then, $\text{Diff}_{A^{s_a} B^{s_b}}$ is the set of vertices that cannot form a supporting pair with u:

$$
\text{Diff}_{A^{s_a} B^{s_b}} = \{v \in V \mid v \in I^{s_a} \wedge v \notin J^{s_b}\}
$$

Depending on the values of s_a and s_b, the cardinality of I^{s_a} and J^{s_b} can easily be computed from the end points of the ranking intervals:

$$
\begin{aligned}
|I^+| &= |\{v \in V | A(v) \leq A(u) \text{ and } v \neq u\}| &= \overline{\rho}(A(u)) - 1 \\
|J^+| &= |\{v \in V | B(v) < B(u) \text{ and } v \neq u\}| &= \underline{\rho}(B(u)) - 1 \\
|I^-| &= |\{v \in V | A(v) \geq A(u) \text{ and } v \neq u\}| &= |V| - \underline{\rho}(A(u)) \\
|J^-| &= |\{v \in V | B(v) > B(u) \text{ and } v \neq u\}| &= |V| - \overline{\rho}(B(u))
\end{aligned}
$$

Figure 3.2 illustrates these sets. In every case, the line represents the vertices sorted by the descriptor depicted on the right, in ascending order. In each line, we distinguish a given vertex u and the end points of the interval containing the vertices with the same value as u ($\underline{\rho}(m(u))$ and $\overline{\rho}(m(u))$). Besides, the hatched gray rectangle gives the set I^{s_a} or J^{s_b}.

Since we cannot derive the exact cardinality of $\text{Diff}_{A^{s_a} B^{s_b}}$, given that we do not know how the sets I^{s_a} and J^{s_b} intersect, we compute a lower bound on it. If $|I^{s_a}| \geq |J^{s_b}|$, then the cardinality of
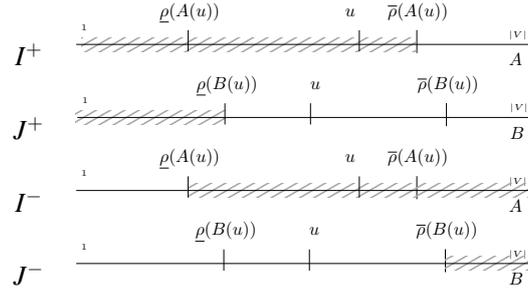
Figure 3.2: Illustration of the computation of $\text{Diff}_{A^{s_a}B^{s_b}}$.

$\text{Diff}_{A^{s_a}B^{s_b}}$ is minimal when $J^{s_b} \subseteq I^{s_a}$. Analogously, if $|I^{s_a}| < |J^{s_b}|$, then $\text{Diff}_{A^{s_a}B^{s_b}}$ can be empty, and thus its cardinality is 0. Thus,

$$|\text{Diff}_{A^+B^+}| \geq \max\{0, \left(\overline{\rho}(A(u)) - \underline{\rho}(B(u))\right)\}$$

$$|\text{Diff}_{A^-B^-}| \geq \max\{0, \left(\overline{\rho}(B(u)) - \underline{\rho}(A(u))\right)\}$$

$$|\text{Diff}_{A^+B^-}| \geq \max\{0, (\overline{\rho}(A(u)) - 1 - (|V| - \overline{\rho}(B(u))))\}$$

$$|\text{Diff}_{A^-B^+}| \geq \max\{0, (|V| - \underline{\rho}(A(u))) - \left(\underline{\rho}(B(u)) - 1\right))\}$$

To establish an upper bound on the support of a pattern P, we take, for each vertex u, the pair of signed descriptors $A^{s_a}B^{s_b}$ of P that maximizes $\text{Diff}_{A^{s_a}B^{s_b}}$: $\text{maxDiff}_P(u) = \max_{A^{s_a}B^{s_b} \in P^2} |\text{Diff}_{A^{s_a}B^{s_b}}|$. For instance, $\text{maxDiff}_{\{i^- \text{BETW}^+\}}(B) = \max\{|\text{Diff}_{i^- \text{BETW}^+}|, |\text{Diff}_{\text{BETW}^+, i^-}|\} = \max\{16 - 7 - 8 + 1, 9 - 1 - 16 + 7\} = 2$. This leads to the following upper bound:

> **Theorem 3.1 — Upper bound on $Supp$.** Let P be a topological pattern,
>
> $$Supp_{all}(P) \leq 1 - \frac{\sum_{u \in V} \text{maxDiff}_P(u)}{n(n-1)} \tag{1}$$

Proof. For each vertex u, let us consider two descriptors A^{s_a} and B^{s_b} from P such as $\text{maxDiff}_P(u) = |\text{Diff}_{A^{s_a}B^{s_b}}(u)|$. This is a lower bound on the number of vertices v such that $(A(v) \trianglerighteq_{s_a} A(u))$ and $\neg(B(v) \triangleright_{s_b} B(u))$. For each such vertex v, neither (u,v) nor (v,u) contributes to $Supp_{all}(P)$. If we sum these numbers over all vertices from V, we get a lower bound on the number of ordered pairs that cannot support P. Since every ordered pair of vertices (u,v) is taken into account twice, we need to divide it by 2 to get a lower bound on the pairs of vertices that do not contribute to the support of P. Finally we divide the upper bound by $\binom{n}{2}$. ∎

Besides, $\text{maxDiff}_P(u)$ is increasing with pattern enumeration, and thus the bound is anti-montone: $\forall P_1 \subseteq P_2, Supp_{all}(P_2) \leq 1 - \frac{\sum_{u \in V} \text{maxDiff}_{P_2}(u)}{n(n-1)} \leq 1 - \frac{\sum_{u \in V} \text{maxDiff}_{P_1}(u)}{n(n-1)}$. Therefore, if $1 - \frac{\sum_{u \in V} \text{maxDiff}_{P_1}(u)}{n(n-1)} \leq \sigma$, all patterns P_2 can be pruned. The constraint $Supp_{all}$ is boundable as define in Definion 1.6.

Observe that this upper bound on $Supp_{all}$ is very convenient since its computation is in $O(|V|)$, whereas the computation of $Supp_{all}$ is in $O(|V|^2)$. On the one hand, it requires storing 2 additional values for every descriptor and every vertex (the end points of the ranking intervals). On the other hand, since we are enumerating descriptors and not descriptor values (as in itemset mining) this is not costly in terms of memory usage.

3.1.2 TopGraphMiner algorithm

TopGraphMiner computes frequent topological patterns and their top k representative vertices from an attributed graph (see Algorithms 4 and 5). It takes in input the graph $G = (V, E, L)$ and two parameters: *minsup* and k. In Line 1 of Algorithm 4, it performs the computation of topological vertex properties. The computation of topological patterns is done in an ECLAT-based way (Zaki 2000). More precisely, all the subsets of a pattern P are always evaluated before P itself. In this way, by storing all frequent patterns in the hash-tree $\mathcal{H}$, the anti-monotonic frequency constraint is fully-checked on the fly (Line 4, in Algorithm 5). We start by enumerating the singleton positive descriptors to avoid the generation of duplicate patterns. Larger patterns are recursively generated by the function EXTEND_PATTERN (see Line 13, in Algorithm 4). We compute the upper bound on the support to prune non-promising topological patterns (function COMP_UB in Line 8 of Algorithm 4). This function is the strict application of Theorem 3.1 (see supplementary material for the pseudo-code). When this upper bound is greater than the minimum threshold, the exact support is computed (function COMP_SUPP in Algorithms 4 and 5).

Another optimization is based on the deduction of the support from already evaluated patterns (function COMP_DEDUC in Line 5 of Algorithm 5). A pair of vertices that supports a pattern P can support pattern PA^+ or pattern PA^-, or none of them. Thus, another upper bound on $Supp_{all}(PA^-)$ is $Supp_{all}(P) - Supp_{all}(PA^+)$. Note that these patterns have already been considered before the evaluation of PA^-. So, to be stringent, we bound the support by taking the minimum between this value and the upper bound defined in Theorem 3.1 (see Line 5 in Algorithm 5). When computing the support of the pattern, the top k representative vertices are also identified.

Algorithm 4 TopGraphMiner

Require: $G = (V, E, L)$, *minsup*, k

Ensure: $\mathcal{H}$: the frequent topological patterns and their top k representative vertices.

1: Compute T, the set of topological properties of G that associate a numerical value to vertices of V based on the relation E.
2:   $\mathcal{M} \leftarrow T \cup L$
3:   $\mathcal{H} \leftarrow \emptyset$
4: **for all** $m \in \mathcal{M}$, in descending order **do**
5:     **for all** $v \in V$ **do**
6:       Compute $\overline{\rho}(m(v))$ and $\underline{\rho}(m(v))$.
7:     **end for**
8:     $UB \leftarrow$ COMP_UB$(\{m^+\}, \overline{\rho}, \underline{\rho})$
9:     **if** $(UB \geq minsup)$ **then**
10:       $(supp, topk) \leftarrow$ COMP_SUPP$(\{m^+\}, k)$
11:       **if** $(supp \geq minsup)$ **then**
12:         $\mathcal{H} \leftarrow \mathcal{H} \cup (\{m^+\}, topk)$
13:         EXTEND_PATTERN $(\{m^+\})$
14:       **end if**
15:     **end if**
16: **end for**

Computation of $Supp_{all}$

The support of P is evaluated by function COMP_SUPP that counts the number of pairs of vertices (u, v) such that $\forall A^{s_a} \in P, A(u) \rhd_{s_a} A(v)$. This computation requires to perform a quadratic operation on the number of vertices. However, as proposed in (Calders, Goethals, and Jaroszewicz

---

Algorithm 5 Extend_Pattern

Require: P a topological pattern, $minsup$, k, $\overline{\rho}$, $\underline{\rho}$
Ensure: Compute all frequent extensions of P and add them to the global variable $\mathcal{H}$ with their
   top k representative vertices

 1: **for all** $B \in \mathcal{M}$, B greater than the last descriptor in P **do**
 2:   **for all** $s \in \{+, -\}$ **do**
 3:     $Q \leftarrow P \cup \{B^s\}$
 4:     **if** $(\forall R \subset Q, R \in \mathcal{H})$ **then**
 5:       $UB \leftarrow \min\{\text{COMP\_UB}(Q, \overline{\rho}, \underline{\rho}), \text{COMP\_DEDUC}(Q, \mathcal{H})\}$
 6:       **if** $(UB \geq minsup)$ **then**
 7:         $(supp, topk) \leftarrow \text{COMP\_SUPP}(Q, k)$
 8:         **if** $(supp \geq minsup)$ **then**
 9:           $\mathcal{H} \leftarrow \mathcal{H} \cup (Q, topk)$
10:           $\text{EXTEND\_PATTERN}(Q)$
11:         **end if**
12:       **end if**
13:     **end if**
14:   **end for**
15: **end for**

2006), a more directed search for all vertices that have smaller or greater values on all descriptors in P is implemented by using range trees and it enable good performances when $|P|$ is not too large. For a singleton pattern $\{m^+\}$, the range tree is simply a binary search tree where each node contains a value x of m along with two values: y^+, that is, the number of vertices that are lower than or equal to x, and y^-, that is, the number of vertices having a value greater or equal to x. Then, to compute the support of $\{m^+\}$, we simply loop over the vertices of the graph, find their corresponding nodes in the range tree and sum the y^+ values of their left subtrees. When extending a pattern P, every node in the range tree is expanded to contain a nested range tree that corresponds to the added descriptor. To compute the support, we loop over the graph vertices, find their corresponding nodes in the inner range trees and sum up the y^+ (resp. y^-) values for positive (resp. negative) descriptors of their left (resp. right) subtrees.

Computation of the top k representatives

As explained in Section 3.1.1, the vertex pairs $S(P)$ that support a topological pattern P define a transitive acyclic directed graph $G_P = (V, S(P))$ (see Property 3.1) that admits at least one topological ordering of its vertices. The top k representative vertices are the k last vertices with respect to one of these orderings.

> **Property 3.3**   Let $G = (V, A)$ be a transitive directed graph and let $Deg^-(v)$ be the incoming degree of the vertex $v \in V$ ($deg^-(v) = |\{\forall u \in V \text{ such that } (u, v) \in A\}|$). For any arc $(u, v) \in A$, $deg^-(u) \leq deg^-(v) + 1$.

Proof. Given an arc $(u, v) \in A$, $\forall t \in V$ such that $(t, u) \in A$, by transitivity of G there exists an arc $(t, v) \in A$. Therefore, $deg^-(u) \leq deg^-(v) + 1$.   ∎

As a result, ordering V with respect to deg^- constitutes a topological sorting of G_P. The range trees used for computing the support of P is exploited to retrieve the top k representative vertices of P: during the loop over the vertices of the graph, their incoming degree is considered

and the set of k vertices having the largest incoming degree is maintained in a heap, using operations in $O(\log k)$.

Computation of $Supp_{A^r}$, $Supp_E$ and Gr

Emerging topological patterns can easily be computed by adapting Algorithm 2: the selected descriptor A^r is the last one in the pattern being enumerated (in the ECLAT enumeration fashion, the last descriptor in the pattern is the first to be enumerated), and when enumerated, its support provides $Supp_{A^r}(P)$. When subtracting this value from the support of its direct ancestor, it provides $Supp_{A^{\bar{r}}}(P)$. We therefore retrieve only those patterns with a growth-rate higher than a threshold. The computation of $Supp_E(P)$ can be done in a time complexity proportional to the number of edges in the graph. Finally, $Gr(P,E)$ can be deduced from $Supp_E(P)$ and $Supp_{all}(P)$.

3.1.3 Experiments

We apply TopGraphMiner on two real-world attributed graphs:

1. DBLP: This co-authorship graph is built from the DBLP digital library. Each vertex represents an author who published at least one paper in one of the major conferences and journals of the Data Mining and Database communities[1] between January 1990 and February 2011. Each edge links two authors who co-authored at least one paper (no matter the conference or journal). The vertex properties are the number of publications in each of the 29 selected conferences or journals.

2. MOVIES: Each vertex of this graph represents a movie and an edge exists between two movies if they have an actor in common[2]. The vertex attributes are based on movie ratings from Netflix customers: the number of ratings, their average and standard deviation values, the release year of the movie and its number of actors.

The main characteristics of these graphs are reported in Table 3.1. All these properties have a minimum value of 0. Many of these properties have a standard-deviation greater than their average, suggesting that they follow power law distributions. The computation of the topological descriptor values in these networks take few hours. For instance, the computation of centrality measures in DBLP, which is the most expensive, takes around 4 hours.

(A) (B) (C)
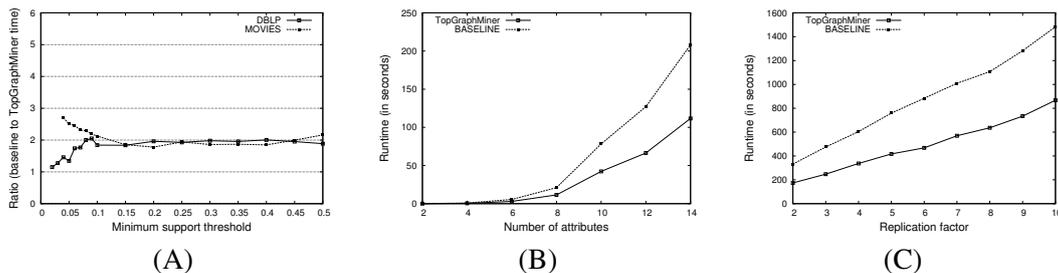
Figure 3.3: Comparison w.r.t. a baseline technique: execution time ratio (A), execution time w.r.t. the number of descriptors (MOVIES, minsup=20%) (B), execution time w.r.t. a replication factor (MOVIES, minsup=20%) (C).

Comparison with a baseline approach

Since there is no other algorithm that simultaneously computes up and down co-variations using the same support measure as in our approach, we first study the performance of TopGraphMiner

---

[1]Conferences: KDD, ICDM, ECML/PKDD, PAKDD, SIAM DM, AAAI, ICML, IJCAI, IDA, DASFAA, VLDB, CIKM, SIGMOD, PODS, ICDE, EDBT, ICDT, SAC – Journals: IEEE TKDE, DAMI, IEEE Int. Sys., SIGKDD Exp., Comm. ACM, IDA J., KAIS, SADM, PVLDB, VLDB J., ACM TKDD.

[2]http://www.imdb.com/

Attributed graph	DBLP			MOVIES		
#Vertices	$42,252$			$5,972$		
#Edges	$210,320$			$64,338$		
#Vertex attributes	29			5		
Density	2×10^{-4}			3.6×10^{-3}		
#Connected Comp.	577			33		
#Communities	1016			56		
Topo. prop.	Max	Mean	Std. Dev.	Max	Mean	Std. Dev.
Raw degree	304	9.73	14.22	118	21.16	19.13
DEGREE	7.3×10^{-3}	2.4×10^{-4}	3.4×10^{-4}	2.2×10^{-2}	4×10^{-3}	3.5×10^{-3}
CLUST	1	0.31	0.29	1.57	0.34	0.26
NBQC	4.6×10^{5}	2.2×10^{2}	7.8×10^{3}	503	2.96	19.93
SZQC	35	2.75	4.83	52	13.87	11.35
SZCOM	$9,342$	40.67	5×10^{2}	$1,563$	11.5×10^{2}	5.6×10^{2}
CLOSE	1	0.024	0.137	1	0.010	0.099
BETW	2.6×10^{6}	1.4×10^{5}	5.7×10^{5}	1.6×10^{5}	1.1×10^{4}	1.6×10^{4}
EGVECT	0.003	2.36×10^{-5}	9.91×10^{-5}	8.4×10^{-3}	1.6×10^{-4}	7.5×10^{-4}
PAGERANK	21.53	0.98	0.98	0.59	0.88	0.59

Table 3.1: Main characteristics of the graphs DBLP and MOVIES

by comparing it with a baseline approach. It consists in using the algorithm of (Calders, Goethals, and Jaroszewicz 2006), which only computes up co-variations, after having duplicate and reverse each descriptor. For instance, the vertex ranked first with respect to the descriptor m^+ is ranked last with respect to m^-. Notice that non-sensible patterns, such as $\{m^+, m^-\}$, will be discarded in linear time since their support is 0. Besides, it is necessary to post-process the output patterns to remove the symmetrical patterns. This additional step is quadratic in the size of the output and can be computationnaly expensive. However, for these experiments we do not take into account the execution time of this post-processing step.

Figure 3.3(A) gives the ratio of the execution time of the baseline approach to the execution time of our approach. We can see that for the graph MOVIES, our approach is at least twice as faster as the baseline. Besides, the lower the support, the higher this ratio is. This behavior shows that our approach is more efficient than the baseline one and that this efficiency does not only rely on the fact that the number of descriptors of the graphs is twice as smaller than the one used by the baseline approach, but also on the pruning capability. With the DBLP graph, however, the ratio decreases for lower supports. This can be explained by the fact that there are many non-frequent topological patterns with negative signs that are early pruned by the baseline approach. Figure 3.3(B) shows the execution time spent by both algorithms with respect to different numbers of randomly chosen original descriptors from the MOVIES graph, with minimum support of 20%. We can observe that our approach outperforms the baseline one and the gain is more important when the number of descriptors increases. Figure 3.3(C) gives the execution time spent by both algorithms with respect to the number of vertices in the attributed graph MOVIES, with minimum support of 20% (the x-axis gives the replication factor). We can notice that TopGraphMiner is faster than the baseline approach and this especially as the number of vertices increases. Although the computation of the support of the patterns is quadratic in the number of vertices, the execution times do not increase accordingly due to the use of the range trees. We can therefore conclude that the results shown in Figure 3.3(A) are more influenced by the number of descriptors than that of vertices.

Tell us where you publish, we tell you how important you are.

We examine the results obtained by TopGraphMiner on the DBLP attributed graph regarding the following questions:

- Are there any interesting patterns among publications?
- Are there interesting trends between some authors' publications and topological properties?
- What about IEEE TKDE authors?

Before extracting topological patterns with TopGraphMiner, we compute correlations between descriptors. The resulting correlation matrix is reported in Figure 3.4(A). The vertex attributes that have a correlation higher than 0.7 are VLDB, ICDE and SIGMOD. The most correlated topological properties are, on the one hand, BETW, DEGREE and PAGERANK and, on the other hand, SZQC and NBQC. The vertex attributes and the topological properties that are not correlated with any other (with a correlation always lower than 0.2) are: SAC, Comm. of ACM, IEEE Int. Sys., CLOSE and CLUST. These correlation measures will help us in the interpretation of the following results.

| (A) | (B) |
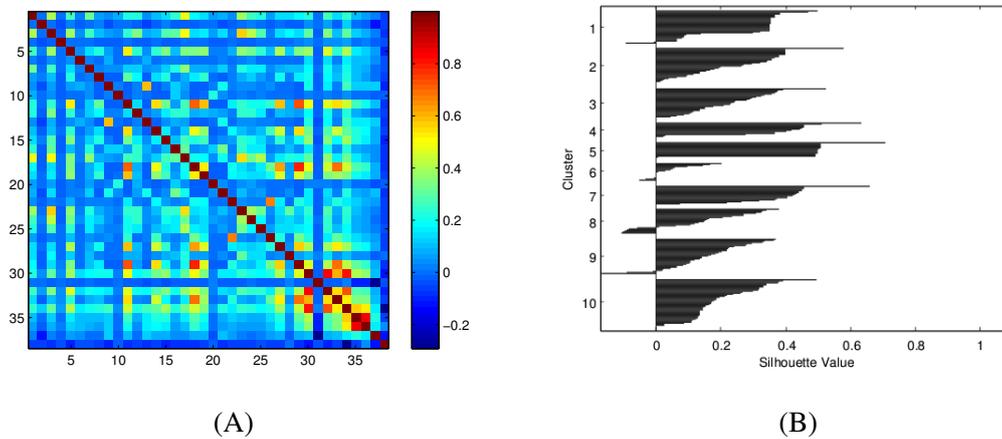
Figure 3.4: Correlation matrix between vertex attributes (1 to 29) and topological properties (30 to 38) in DBLP (A). Silhouette plot of the K-means clustering on some topological patterns (B).

Topological patterns on conferences and journals: Let us first consider topological patterns among publications venues. Mining all frequent topological patterns with a support threshold of 1% takes 68 seconds. The output contains 263 topological patterns, from which 58 (22%) involve negatively signed attributes. To better understand the type of information retrieved by these 263 patterns, we performed a clustering analysis of the topological patterns. We use K-means algorithm on the 263×57 Boolean matrix where the rows correspond to the patterns and the columns to the signed vertex attributes ($2 \times 29 - 1$). We use the cosine distance and employ the silhouette plot to determine the number of clusters. It suggests 10 clusters (see Figure 3.4(B)). The most frequent vertex attributes of each cluster are shown in Table 3.2, that is the vertex attributes that appear in at least in 2/3 of the cluster patterns. We can observe that the majority of the clusters are homogeneous, referring either to Data mining or to Database publications. For instance, Clusters 1, 2, 6, and 9 refer to Data mining publications, while Clusters 3, 8, and 10 clearly refer to Database publications. Other clusters are related to a specific conference/journal.

Interestingly, 20 of these patterns contain the attribute SAC^- together with positively signed attributes. Examples of such patterns are $\{SAC^-, KDD^+\}$, $\{SAC^-, ECML/PKDD^+\}$, $\{SAC^-, VLDB^+\}$, and $\{SAC^-, SIGMOD^+\}$. This type of pattern can be explained by the fact that SAC scope is larger than that of the other selected conferences, which are more focused either on

Cluster	# patterns	Most frequent vertex attributes	Cluster	# patterns	Most frequent vertex attributes
1	32	SAC^-, $IJCAI^+$	6	17	$KAIS^+$, SDM^+, $PAKDD^+$, KDD^+
2	34	$CIKM^+$, $PAKDD^+$	7	18	$IEEE\ TKDE^+$
3	28	$SIGMOD^+$	8	24	$VLDB^+$, $VLDBJ^+$, $PVLDB^+$
4	15	$AAAI^+$	9	34	$ICDM^+$, $PKDD^+$, KDD^+
5	15	$CommACM^+$	10	46	$ICDE^+$, $SIGMOD^+$, $TKDE^+$, $VLDB^+$

Table 3.2: Most frequent vertex attributes in clusters of patterns found in the DBLP attributed graph.

Database or Data Mining topics. Since the topics covered by SAC are much more general (e.g., Programming Languages, Geometric Constraints and Reasoning, and Applied Biometrics), it is not surprising that many authors that have several publications in SAC conference series have none or few publications in the Data Mining or Database areas.

Are there interesting trends between author publications and topological properties?
Table 3.3 reports the most frequent pattern (P_{all}), the most emerging pattern (P_{PAGERANK^+}) with respect to PAGERANK^+ and the most structurally correlated topological pattern (P_E). P_{all} is formed by descriptors SAC^+ and $SZCOM^-$. Its meaning is that SAC authors tend to belong to small communities, that is, these authors are rather isolated in the graph as illustrated in Figure 3.5(A), where the top-10 representative vertices and their direct neighborhoods are displayed. These vertices have a low degree. As mentioned before, the scope of the SAC conference is much wider than Database and Data mining topics. This makes this pattern sensible and justifies that (1) this pattern is not much correlated to the graph structure ($Gr(P,E) = 0.21$), and (2) its top-5 supporting vertices are mostly researchers from Software engineering and Network areas.

P	Descriptors	Measures	Top-5 Representative vertices
P_{all}	SAC^+, $SZCOM^-$	$Supp_{all} = 0.19$ $Gr(P,E) = 0.21$	#1 F. N. Sibai, #2 M. M. Huntbach, #3 C. Leopold, #4 A. J. Duben, #5 P. Rittgen,
P_{PAGERANK^+}	$ICDE^+$, $DEGREE^+$, $BETW^+$, $CLUST^-$, $NBQC^+$, $SZQC^+$	$Gr(P, \text{PAGERANK}^+) = 253,933$ $Gr(P,E) = 4.8$ $Supp_{all} = 0.12$	#1 H. Garcia-Molina, #2 M. Stonebraker, #3 G. Weikum #4 R. Agrawal, #5 M. J. Franklin,
P_E	$PVLDB^+$, $DEGREE^+$, $BETW^+$	$Gr(P,E) = 6.9682$	#1 G. Weikum, #2 J. Han, #3 D. Maier #4 P. S. Yu, #5 H. Garcia-Molina,

Table 3.3: Top topological patterns in the DBLP attributed graph.

The computation of emerging patterns with respect to PAGERANK, with a support threshold of 1% and a growth-rate threshold of 3, takes around 6 hours and produces $4,313$ patterns. The most emerging pattern P_{PAGERANK^+} (see Table 3.3) contains many topological properties with a positive sign, except $CLUST$, which has a negative sign. As we have seen before, PAGERANK is

highly correlated with DEGREE and BETW. Therefore, it is not surprising that both appear in the pattern. On the other hand, the presence of the property CLUST$^-$ suggests that the higher the PAGERANK of the authors (and consequently their DEGREE and BETW), the lower the connectivity of their co-authors. In other words, authors with high PAGERANK have many co-authors that do not publish together. This can be observed on Figure 3.5(B) where the connectivity between co-authors of the top-10 representative vertices is low. Those that advise many PhD students can be seen as typical examples of these authors.

The most structurally correlated topological pattern P_E gathers the descriptors PVLDB$^+$, DEGREE$^+$ and BETW$^+$. PVLDB is at the same time a well-established conference and journal in the Data mining and Database communities. This pattern is strongly structurally correlated ($Gr(P,E) > 5$), i.e., it tends to be more supported by pairs that are edges than arbitrary pairs of vertices. Figure 3.5(C) displays its top-10 representative vertices.

(A)   (B)   (C)
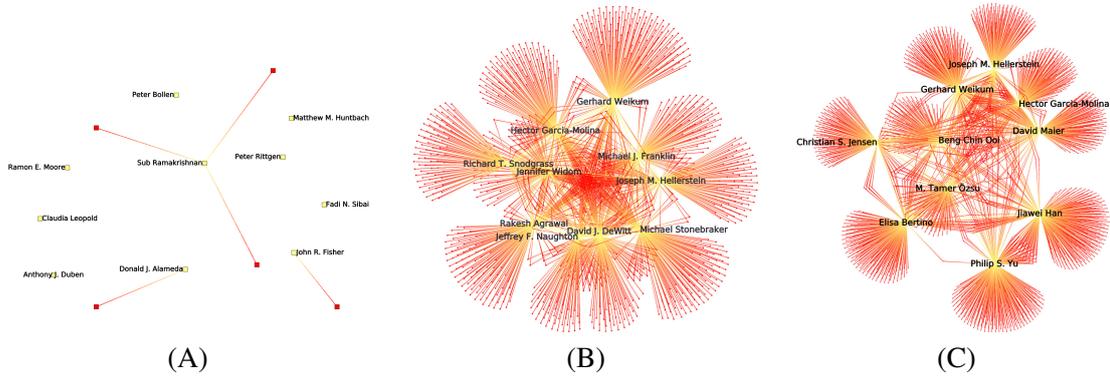
Figure 3.5: Top 10 vertices supporting P_{all} (A), $P_{PAGERANK}$ (B) and P_E (C) and their connected vertices in DBLP.

We can also use emerging topological patterns, made only of topological properties, to compare the relative importance of conferences and journals. Let us consider $P_{Topo_1} = \{$PAGERANK$^+$ DEGREE$^+\}$ and $P_{Topo_2} = \{$PAGERANK$^+$, BETW$^+\}$, two such emerging patterns whose respective growth-rates are $Gr(P_{Topo_1}, $PAGERANK$^+) = 124.69$ and $Gr(P_{Topo_2}, $PAGERANK$^+) = 584.46$. These emerging patterns reveal which conferences or journals are more related to the topological properties BETW$^+$ and DEGREE$^+$. To that end, for each publication venue C and both emerging patterns P_{Topo_1} and P_{Topo_2}, we compute the ratio $\frac{Gr(P_{Topo_i}C, \text{PAGERANK}^+)}{Gr(P_{Topo_i}, \text{PAGERANK}^+)}$. Table 3.4(A) gives the top-5 publications with respect to this ratio. Surprisingly, we observe that Data Mining conferences have a higher impact on the pattern $\{$PAGERANK$^+$, DEGREE$^+\}$, while Database conferences positively influence the growth-rate of the pattern $\{$PAGERANK$^+$, BETW$^+\}$. Since Data Mining intersects many other research areas, these results may be explained by the fact that Data Mining authors may also publish with many others from different areas, such as Database and Machine Learning ones. On the other hand, as Database is an older well-established research field, Database authors tend to appear at the center of the graph. For the most impacting publications, we identify the top-5 representative authors. They are shown in Table 3.4(B).

What about the IEEE TKDE authors? We also look for the emerging patterns with respect to the attribute IEEE TKDE, with support threshold of 1% and growth-rate threshold of 3 (their computation takes around 5 hours). We obtain 745 emerging patterns with respect to the class IEEE TKDE$^+$. The most emerging pattern is $P_{TKDE} = $ ICDE$^+$, VLDB$^+$, BETW$^+$, PAGERANK$^+$, with $Gr(P_{TKDE}, $TKDE$^+) = 11.75$. This pattern indicates that authors publishing in IEEE TKDE journal tend also to publish papers in the conferences ICDE and VLDB. BETW$^+$ suggests that these authors are located at the center of the co-authorship graph, while PAGERANK$^+$ means that they co-

Rank	P_{Topo_1}		P_{Topo_2}		PAGERANK$^+$ DEGREE$^+$ ECML/PKDD$^+$	PAGERANK$^+$ BETW$^+$ PVLDB$^+$
	Publication	Factor	Publication	Factor		
1	ECML/PKDD$^+$	2.5	PVLDB$^+$	5.67	Christos Faloutsos	Gerhard Weikum
2	IEEE TKDE$^+$	2.28	EDBT$^+$	5.11	Jiawei Han	Jiawei Han
3	PAKDD$^+$	2.21	VLDB J.$^+$	4.35	Philip S. Yu	David Maier
4	DASFAA$^+$	2.09	SIGMOD$^+$	4.25	Bing Liu	Philip S. Yu
5	ICDM$^+$	1.95	ICDE$^+$	3.42	C. Lee Giles	Hector Garcia-Molina
	(A)				(B)	

Table 3.4: Top-5 "impacting" publications in the emergence of DEGREE$^+$ and BETW$^+$ w.r.t. PAGERANK$^+$ (A) along with their top-5 authors (B).

authored papers with other researchers that also appear at the center of the graph. It is important to observe that this pattern is also highly structurally correlated ($Gr(P_{\text{TKDE}}, E) = 6.5758$). Furthermore, this pattern is sensible since it is supported by well-established researchers in the Database community: Christos Faloutsos, Jiawei Han, Philip S. Yu, Beng Chin Ooi, and Hector Garcia-Molina are its top-5 representative authors.

Do we only appreciate blockbusters?

Let us now consider the real-world attributed graph MOVIES. Table 3.5 shows the 4 most frequent topological patterns (with at least 2 descriptors) with their top-5 representative movies. Pattern

P	Descriptors	Measures	Top-5 movies
P_1	AVG_RATING$^+$ NB_RATINGS$^+$	$Supp_{all} = 0.7$ $Gr(P,E) = 1.05$	#1 The Green Mile, #2 Forrest Gump, #3 The Sixth Sense, #4 Indiana Jones and the Last Crusade, #5 Gladiator
P_2	NB_RATINGS$^+$ CLOSE$^+$	$Supp_{all} = 0.6$ $Gr(P,E) = 0.87$	#1 The Rock, #2 Fahrenheit 9/11, #3 The Godfather, #4 Enemy of the State, #5 Men of Honor
P_3	STD_RATING$^+$ PAGERANK$^-$	$Supp_{all} = 0.58$ $Gr(P,E) = 0.89$	#1 There's no Business Like Show Business, #2 Michael Moore Hates America, #3 Digimon: The Movie, #4 Blown Away, #5 Benjamin Smoke
P_4	YEAR$^+$ AVG_RATING$^-$	$Supp_{all} = 0.57$ $Gr(P,E) = 0.94$	#1 Day of the Dead 2: Contagium, #2 raging sharks, #3 My Big Phat Hip Hop Family, #4 The Fallen Ones, #5 Last Days

Table 3.5: Patterns found in MOVIES and their top-5 movies.

P_1 suggests that Netflix users tend to rate movies they like. Its top-10 representative movies are connected (see Figure 3.6(A)), which indicates they have at least one actor in common. The second pattern P_2 reveals that many users tend to rate movies located at the center of the graph, that is, movies with "major" actors (e.g., R. de Niro, S. Connery, T. Hanks, B. Willis, H. Ford, etc.). Therefore, the supporting vertices of this pattern is made of major blockbusters (see Figure 3.6(B)). Pattern P_3 indicates that controversial movies (those with a high rating standard deviation) tend to be isolated within the graph (lower PAGERANK): they are more independent films without well-known actors. Note that all the supporting movies of this pattern have a degree of 0. Finally, pattern P_4 suggests that older movies are better rated. This can be due to the fact that the ratings were given between 1998 and 2005. Therefore, Netflix users tend to rate only non-contemporary movies they like and to forget those they did not like over time.

Table 3.6 shows the most emerging topological pattern with respect to the PAGERANK and the most structurally correlated pattern. Pattern P_{PAGERANK} gathers descriptors NB_ACTORS$^+$ and all the centrality measures, plus STD_RATING$^-$ and NB_RATINGS$^+$. As the edges of MOVIES encode the fact that two movies share at least one actor, it is not surprising that this pattern associates NB_ACTORS$^+$ with all centrality measures. Furthermore, the attribute STD_RATING$^-$ indicates that the representative movies of this pattern are consensual.

The most structurally correlated topological pattern P_E reveals that recent movies (YEAR$^+$) tend to play a central role within the graph (BETW$^+$, EGVECT$^+$, PAGERANK$^+$) and their neighbors

P	Descriptors	Measures	Top-5 movies
P_{PAGERANK^+}	NB_ACTORS$^+$ STD_RATING$^-$ NB_RATINGS$^+$ DEGREE$^+$ CLOSE$^+$ BETW$^+$ EGVECT$^+$ NBQC$^+$ SZQC$^+$ SZCOM$^+$	$Supp_{all} = 0.052$ $Gr(P,\text{PAGERANK}^+) = 11,789$ $Gr(P,E) = 0.32$	#1 The Godfather, #2 Crimson Tide, #3 The Untouchables, #4 The Hunt for Red October, #5 Apollo 13,
P_E	YEAR$^+$, BETW$^+$, EGVECT$^+$, PAGERANK$^+$, CLUST$^-$	$Supp_{all} = 0.05$ $Gr(P,E) = 2.78$	#1 Catch me if you can, #2 True Crime, #3 Batman Begins, #4 The Quiet American, #5 Scenes of the Crime,

Table 3.6: Top topological patterns in MOVIES.

tend to be not connected (CLUST$^-$), since it is not common that several movies share the same casting. The projection of its top-10 representative vertices on the graph is given in Figure 3.6(C).

(A) (B) (C)
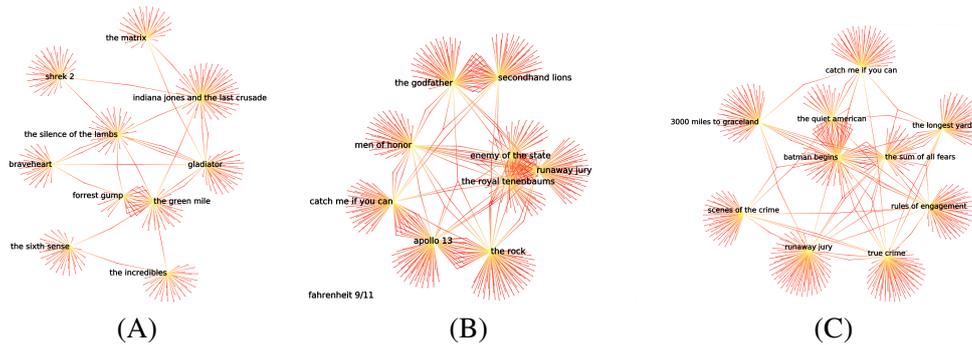
Figure 3.6: Top 10 vertices supporting P_1 (A), P_2 (B), and P_E (C) and their connected vertices in MOVIES.

The previous results show the capability of TopGraphMiner to discover sensible patterns.

3.2 Evolution patterns in dynamic graphs

As real-life graphs tend to change dynamically, several proposals consider the extraction of patterns in dynamic graphs. Borgwardt, Kriegel, and Wackersreuther 2006 introduces the problem of mining frequent sub-graphs in dynamic graphs, that is to say isomorphic graphs that appears in consecutive time steps. Lahiri and Berger-Wolf 2008 also extracts frequent sub-graphs but with a periodic appearance. Böttcher, Höppner, and Spiliopoulou 2008 proposes a knowledge discovery framework to detect and analyze when and how changes occur from a non-stationary population. Giannotti, Nanni, Pinelli, and Pedreschi 2007 aims to mine spatio-temporal data by computing concise descriptions of frequent behaviors.

In Robardet 2009, I propose to capture the evolution of dynamic graphs by extracting evolving patterns as pseudo-cliques which appears in consecutive timestamps with slight modifications. We designed a local to global approach (Raedt and Zimmermann 2007) that proceeds in two steps: It first computes local patterns that capture locally strong associations between vertices at each time step, and second, models the dynamics of the graph as the evolution in time of these patterns. Local patterns, defined by a set of inductive constraints, are extracted by means of an exploratory analysis of the search-space where the validity of each local pattern is evaluated independently from the other ones. To identify the graph areas of interest, we compute specific sub-graphs, defined as isolated pseudo-cliques. Pattern sets of isolated pseudo-cliques, that satisfy some global constraints, capture the dynamics of the graph. The constraints do no longer rely on a single candidate pattern but depend on the sets of isolated pseudo-cliques extracted at two consecutive time steps. Five types of temporal patterns are considered that makes possible the identification of isolated pseudo-cliques that grow, diminish, disappear, emerge or be stable. These evolving patterns allow us to describe the processes by which communities come together, attract new members, and develop over time.

3.2.1 Constraint-based sub-graphs in static graphs

Let $G = (V, E)$ be a static undirected graph, where V is the set of vertices and E, $E \subseteq V \times V$, the set of edges. 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 sub-graph induced by a subset of vertices S ($S \subseteq V$) is the graph $G_S = (S, E_S)$ where $E_S = \{\{u, v\} \mid \{u, v\} \in E \text{ and } u, v \in S\}$.

Sub-Graphs of interest are usually those made of vertices that have a high density of edges. The density of a sub-graph is defined as the number of edges between vertices of the sub-graph, divided by the maximal number of possible edges. A clique is a sub-graph whose density equals 1. To relax this strong property, we consider pseudo-cliques defined as having a density at least equal to a user-defined threshold σ.

> **Definition 3.4 — Pseudo-clique constraint.** Given a user-defined threshold $\sigma \in [0, 1]$ and a set of vertices $S \subseteq V$ of size n, the sub-graph $G_S = (S, E_S)$ 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
>
> $$\mathscr{C}_{pc}(S, \sigma) \equiv \frac{\sum_{u \in S} \deg_S(u)}{n(n-1)} \geq \sigma \tag{2}$$

The pseudo-clique constraint is not anti-monotone with respect to the enumeration of sub-graphs based on the set inclusion of their vertex sets: Expanding a set S of n vertices could make the density $\left(\frac{2|E_S|}{n(n-1)} \right)$ increases or decreases. 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 (Zhu, Yan, Han, and Yu 2007).

> **Property 3.4 — Loose anti-monotone pseudo-clique constraint.** Pseudo-clique constraint is loose anti-monotone, i.e., $\mathscr{C}_{pc}(S, \sigma) \Rightarrow \exists v \in S \text{ such that } \mathscr{C}_{pc}(S \setminus \{v\}, \sigma)$.

Proof. Suppose the sub-graph S satisfies $\mathscr{C}_{pc}(S, \sigma)$. Let $v^\star$ be a vertex of S having the smallest degree on S, i.e. $\deg_S(v^\star) = \min_{u \in S} \deg_S(u)$. Let n be the size of S and $S^\star = S \setminus \{v^\star\}$. Then, we have

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

- 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 $\mathscr{C}_{pc}(S^\star, \sigma)$ 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 $\mathscr{C}_{pc}(S^\star, \sigma)$ is also satisfied. ∎

To be efficient, the enumeration process must take advantage of the pruning power from the loose anti-monotone property of pseudo-cliques. It is clear, from the proof of Property 3.4, that adding to a pseudo-clique S the vertex $v \in V \setminus S$ that satisfies

$$\deg_{S \cup \{v\}}(v) = \min_{u \in S \cup \{v\}} \deg_{S \cup \{v\}}(u) \tag{3}$$

leads to a pseudo-clique, unless none of the supersets of S is a pseudo-clique. Thus, an efficient algorithm enumerates vertices recursively by finding at each iteration the vertex[3] v that satisfies Equation (3) and stop the enumeration if $\mathscr{C}_{pc}(S \cup \{v\}, \sigma)$ 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.

Uno 2007 proposes a method to solve Equations (2) (on $S \cup \{v\}$) and (3) efficiently. As $\sum_{u \in S \cup \{v\}} \deg_{S \cup \{v\}}(u) = \sum_{u \in S} \deg_S(u) + 2 \deg_S(v)$, $\mathscr{C}_{pc}(S \cup \{v\}, \sigma)$ is satisfied if and only if $\deg_S(v) \geq \frac{\sigma |S|(|S|+1) - \sum_{u \in S} \deg_S(u)}{2}$. Equation (2) is thus checked in constant time if $\sum_{u \in S} \deg_S(v)$ is stored and updated during the enumeration process. Equation (3) can trivially be checked in $(O(|V|)$, and less naively in time $O(\deg_S(v))$ (Uno 2007).

Pseudo-cliques capture strong – but not necessarily perfect – associations in a graph. However, not all the pseudo-cliques of a graph are of importance: Some of them have many links to external vertices. To identify the most useful pseudo-cliques, we consider another constraint that coerces the patterns to be isolated. The isolation constraint imposes a maximum to the average number of external links per vertex.

> **Definition 3.5 — Isolated constraint.** Given a user defined threshold $\gamma \in \mathbb{R}$, a sub-graph S is isolated iff
>
> $$\mathscr{C}_i(S, \gamma) \equiv \frac{\sum_{u \in S} (\deg(u) - \deg_S(u))}{|S|} \leq \gamma.$$

> **Property 3.5** $\mathscr{C}_i(S, \gamma)$ is loose anti-monotone.

Proof. Suppose the sub-graph S satisfies $\mathscr{C}_i(S, \gamma)$. Let v be a vertex of S such that

$$v = \arg \max_{u \in S} (\deg(u) - \deg_S(u))$$

Then,

- If $\deg(v) - \deg_S(v) < \gamma$, then $\forall u \in S$, $\deg(u) - \deg_S(u) \leq \deg(v) - \deg_S(v)$ and $\sum_{u \in S \setminus \{v\}} (\deg(u) - \deg_S(u)) < (n-1)\gamma$, i.e., $\mathscr{C}_i(S \setminus \{v\}, \gamma)$ is satisfied.
- Otherwise, $\sum_{u \in S} (\deg(u) - \deg_S(u)) - (\deg(v) - \deg_S(v)) \leq n\gamma - \gamma = (n-1)\gamma$ and $\mathscr{C}_i(S \setminus \{v\}, \gamma)$ is also satisfied.

∎

However, the combination of two loose anti-monotone constraints (Definitions 3.4 and 3.5) is not necessarily loose anti-monotone. To enumerate isolated pseudo-cliques in a single process, the algorithm must find the vertex v that satisfies both Equation (3) and $v = \arg \max_{u \in S \cup \{v\}} \left(\deg(u) - \deg_{S \cup \{v\}}(u) \right)$. The conjunction of these equations characterizes the vertex leading to an isolated pseudo-clique. Such a vertex does not necessarily exist and thus $\mathscr{C}_{pc} \wedge \mathscr{C}_i$ is not loose anti-monotone. Hence, we propose to use $\mathscr{C}_i$ in post-treatment of the computed pseudo-cliques.

3.2.2 Mining evolving sub-graphs

The previous step provides unstructured and numerous patterns. These results are hence difficult (if not impossible) to interpret (Raedt and Zimmermann 2007). 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 dynamics. Let $\mathscr{G} = (G_1, \ldots, G_T)$ be a time-series of graphs,

---

[3]Note that if several vertices satisfy Equation (3), the one of smallest index is taken.

where $G_t = (V, E_t)$ is the graph at time step t, $E_t \subseteq V \times V$. 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 $\mathscr{P}_t$ the set of sub-graphs of G_t that satisfy $\mathscr{C}_{pc} \wedge \mathscr{C}_i$. We consider the five following global constraints:

Stability: S is said to be stable at time t if it is a valid pseudo-clique at both times t and $t-1$:

$$\mathscr{C}_{stable}(S, \mathscr{P}_t, \mathscr{P}_{t-1}) \equiv S \in \mathscr{P}_t \text{ and } S \in \mathscr{P}_{t-1}$$

Growth: A sub-graph S grows 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 $t-1$:

$$\mathscr{C}_{growth}(S, \mathscr{P}_t, \mathscr{P}_{t-1}) \equiv S \in \mathscr{P}_t \text{ and } \exists R \in \mathscr{P}_{t-1}, such that R \subset S$$

Diminution: A sub-graph S diminishes 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 $t-1$:

$$\mathscr{C}_{diminution}(S, \mathscr{P}_t, \mathscr{P}_{t-1}) \equiv S \in \mathscr{P}_t \text{ and } \exists R \in \mathscr{P}_{t-1}, such that S \subset R$$

Extinction: A sub-graph 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:

$$\mathscr{C}_{extinction}(S, \mathscr{P}_t, \mathscr{P}_{t-1}) \equiv S \in \mathscr{P}_{t-1} \text{ and } \forall R, R \subseteq S, R \notin \mathscr{P}_t \text{ and } \forall R, S \subseteq R, R \notin \mathscr{P}_t$$

Emergence: A sub-graph 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}:

$$\mathscr{C}_{emergence}(S, \mathscr{P}_t, \mathscr{P}_{t-1}) \equiv S \in \mathscr{P}_t \text{ and } \forall R, R \subseteq S, R \notin \mathscr{P}_{t-1} \text{ and } \forall R, S \subseteq R, R \notin \mathscr{P}_{t-1}$$

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 (Bodon 2005), 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 sub-graphs 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. Here it is used to retrieve the vertex sets that define valid sub-graphs.

Suppose that pseudo-cliques of $\mathscr{P}_{t-1}$ are stored in a trie $\mathscr{T}$. Each node of $\mathscr{T}$ consists of the set S of vertices of the pseudo-clique, a list of temporal states, a list of pointers to other trie nodes and a list of time steps. When a new valid pseudo-clique of G_t is computed, its vertex set S is inserted in $\mathscr{T}$ 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 step is set to t.
- In the latter case, the node gets a new son with vertex set S, time step 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 $\mathscr{T}$ 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 $\mathscr{T}$ with time step 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 to be considered 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 $\mathscr{P}_{t-1}$) 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 $\mathscr{P}_t$ have been processed. For all the nodes with time step $t-1$ 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 $\mathscr{P}_t$, 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.

3.2.3 Experimental results

We evaluate this approach on three real-world dynamic networks: two dynamic sensor networks, IMOTE and MIT (see Section 2.1.1 for their presentation) and a dynamic mobility network Vélo'v, the shared bicycle system of Lyon (see Section 2.2). The main characteristics of these datasets are presented on Table 3.7.

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

Table 3.7: Dataset characteristics

Dynamic sensor networks
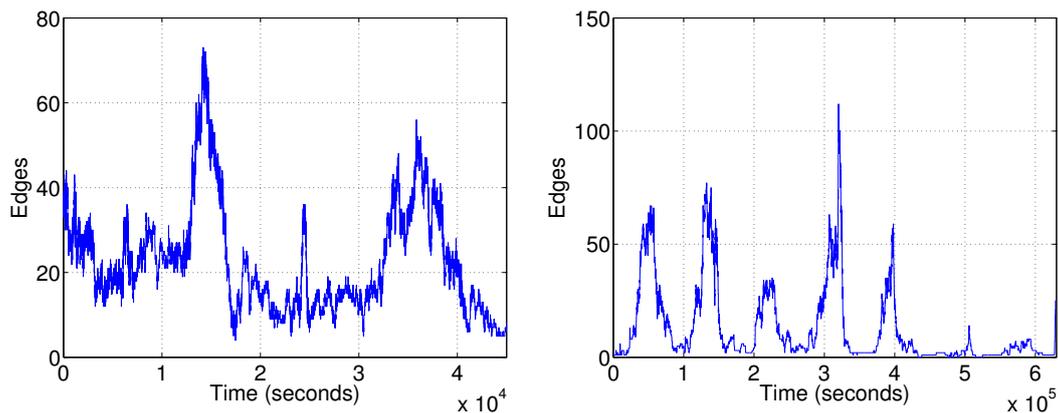
Figure 3.7: Number of edges, displayed as a function of time (IMOTE on the left and MIT on the right).

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

We extract evolving sub-graphs with several density values σ, the isolated parameter γ being set to 4.5 and the minimal size of the extracted valid pseudo-cliques being set to 4 for IMOTE and to 3 for MIT. The total runtimes and number of computed patterns are shown on Figure 3.8 (left). These figures show that the method 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 as it was expected according to the theoretical study of the time complexity of the pseudo-clique mining algorithm discussed in Section 3.2.1. The time required to compute evolving patterns generally decreases with σ as well as the number of extracted patterns.
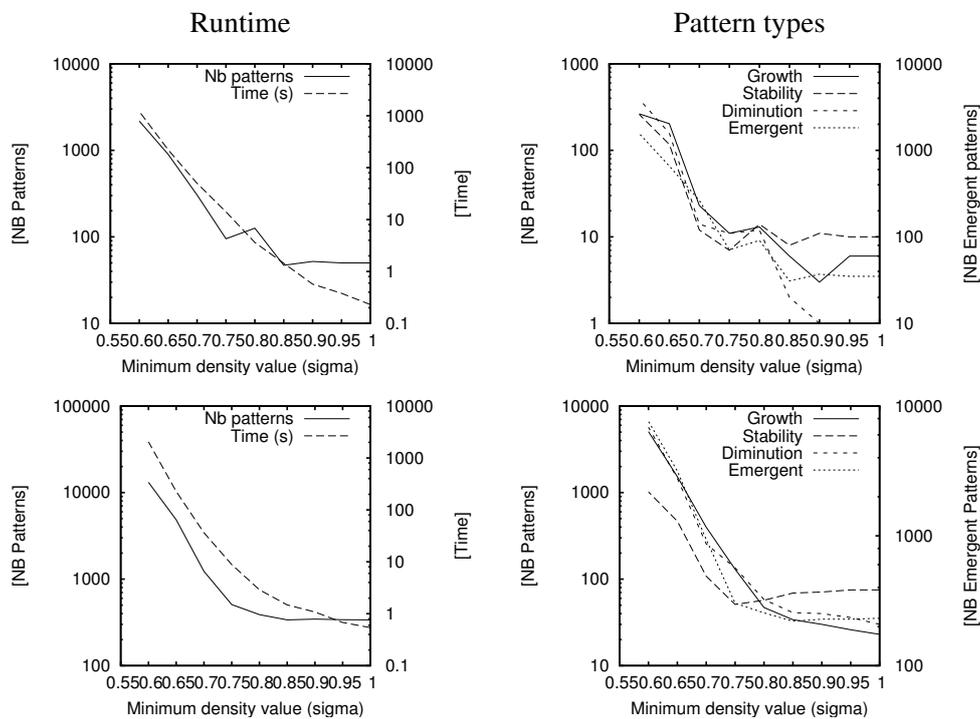
Figure 3.8: Runtime and number of extracted patterns (logarithmic scales) (left) and number of patterns of each type (right) for IMOTE (top) and MIT (bottom) dynamic graphs for different density threshold σ.

The numbers of evolving patterns of each type are shown on Figure 3.8 (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 3.9 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 sub-graphs 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 step 71, splits around time step 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
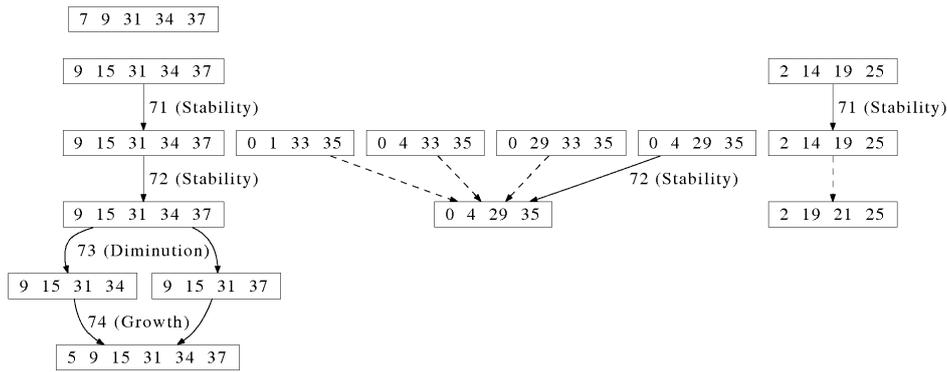
Figure 3.9: Display of the evolving patterns for IMOTE with $\sigma = 0.8$, $\gamma = 3$ and the minimum sub-graph size equals 4 that occur in the morning.

and 33 are nearby. This group is stable since it remains unchanged during two consecutive time steps. The third group contains individuals 2, 14, 19 and 25 and is also stable.

Shared bicycle system Vélo'v

Let us now analyze Lyon's shared bicycle system Vélo'v. 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 steps. Then, to leverage the most important links, we remove the edges that had less than 50 rentals over this period.
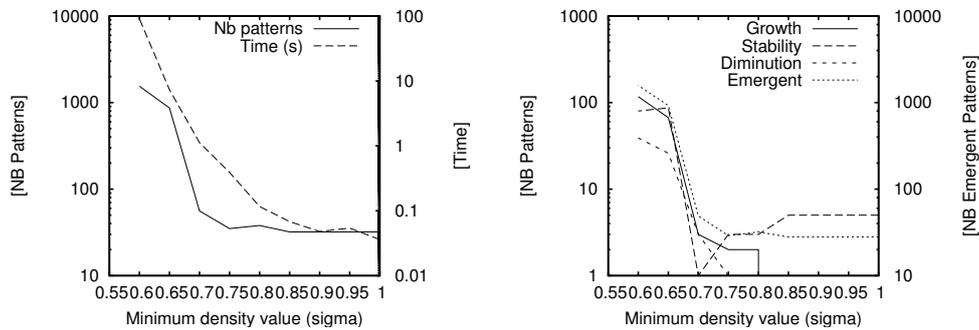
Figure 3.10: Runtime and number of extracted patterns for Vélo'v dynamic networks for different density threshold σ (left), number of patterns of each type (right).

Figure 3.10 (left) shows the total number of extracted patterns and the runtime for several σ values, γ being set to 5 and the minimum sub-graph size to 3. We obtain similar results than those on Figure 3.8: Here again, we can observe that the number of extracted patterns increases with σ. Figure 3.10 (right) shows the repartition of the patterns among the different types of evolving patterns. The majority of the extracted patterns are emergent. The number of identical patterns can increase or decrease with σ: When a stable pattern disappears, usually a growth or diminution pattern appears.

Figure 3.11 displays the output of the method on the Vélo'v dynamic graph for time steps between Monday 6 PM and Tuesday 7 AM. The analysis of this outcome carries interesting pieces of information: For example, at time step 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
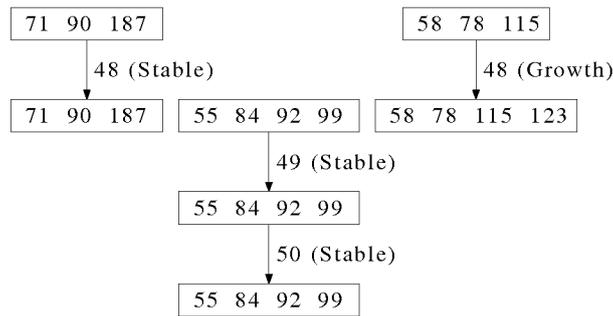
Figure 3.11: Display of the evolving patterns for Vélo'v with $\sigma = 0.9$, $\gamma = 5$ and the minimum sub-graph size equals 3.

123. At the same time another sub-graph 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 3.12. This phenomenon can be related to the closure of the public transports right after midnight: Public transport users, who travel at night like students, become Vélo'v users when public transports are closed. This increases the nocturnal activity of the Vélo'v system.
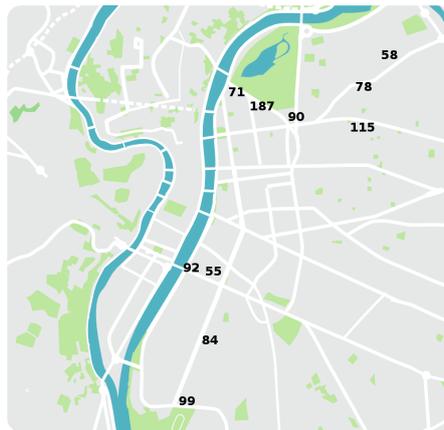
Figure 3.12: Location of Vélo'v stations in Lyon. Patterns from Figure 3.11 are shadowed on the map.

3.3 Trend mining in attributed dynamic graphs

Having proposed data mining methods dedicated to the analysis of attributed graphs and of dynamic graphs, we consider in Desmier, Plantevit, Robardet, and Boulicaut 2013 a technique for extracting valuable information from dynamic attributed graphs. The simultaneous consideration of the graph structure, the vertex attributes and their evolution through time makes possible to tackle a wide variety of mining problems (R. G. Pensa, Cordero, Rouveirol, Kanawati, Troyano, and Rosso 2010). For example, Boden, Günnemann, and Seidl 2012 proposes to extract clusters in each static attributed graph and to associate time consecutive clusters that are similar. Piatkowski, Lee, and Morik 2013 investigates the use of discrete probabilistic graphical models for predicting vertex attributes in a spatio-temporal graphs. Our trend mining approach combines aspects of the methods presented in the two previous Sections. As in the approach presented in

Section 3.1, we consider the evolution of the numerical attributes. But instead of comparing the numerical values of pairs of vertices, we compare here the numerical values of a single vertex at two consecutive time steps. In a manner similar to the approach presented in Section 3.2, we extract specific sub-graphs defined by means of constraints. But whereas in Section 3.2 we compute highly connected sub-graphs, that is to say pseudo-cliques, we consider here sub-graphs constrained by a maximum diameter value that limits the length of the longest shortest path between any two vertices. Indeed, we observed (Desmier, Plantevit, Robardet, and Boulicaut 2012) that the pseudo-clique constraint is very stringent and tends to fragment some reliable patterns. Additional interestingness measures are used to assess the interest of the trend dynamic sub-graphs and guide their search by user-parameterized constraints. These constraints aim at answering the following questions:

- How similar are the vertices outside the trend dynamic sub-graph to the ones inside it?
- Are trends specific to the vertices of the pattern?
- What about the dynamic of the pattern? Does it appear suddenly or continuously?

These constraints are monotone, anti-monotone or piecewise monotone that make possible to use the generic algorithm presented in Section 1.4.

3.3.1 Trend Dynamic Sub-graphs

The input of our mining task is an attributed dynamic graph $\mathscr{G} = \{G_t \mid t = 1 \ldots t_{\max}\}$ over a discrete time span $T = [\![1, t_{\max}]\!]$. Each static graph is a non-directed attributed graph $G_t = (V, E_t, A)$ where V is a set of n vertices $\{v_1, \ldots, v_n\}$ that is fixed throughout the time, $\{E_t \mid t \in T\}$ is a sequence of sets of edges that connect vertices of V at time t ($E_t \subseteq V \times V$), and A is a set of p ordinal attributes $\{a_1, \ldots, a_p\}$ whose values are defined for each vertex at each time step ($a_i : V \times T \to \mathbb{D}_i$, where $\mathbb{D}_i$ is the domain of a_i).

The language of patterns $\mathscr{L}$ considered here is the one of sub-graphs induced from $\mathscr{G}$ by a subset of vertices of V, a sub-sequence of time steps from T and a set of trend attributes. We say that a vertex follows an increasing trend over attribute a at time t, denoted a^+, if $a(u,t) < a(u,t+1)$. In a similar way, a decreasing trend, a^- is characterized by $a(u,t) > a(u,t+1)$. Given a subset of vertices $U \subseteq V$ and a sub-sequence $S = \langle t_1, \cdots, t_s \rangle$ of time steps of T, the dynamic sub-graph of $\mathscr{G}$ induced by (U,S) is $\mathscr{G}(U,S) = \{G_t(U) \mid t \in S\}$ and $G_t(U)$ contains all the edges in E_t that have both ends in U. Therefore, the language of patterns is

$$\mathscr{L} = \{(\mathscr{G}(U,S), \Omega) \mid U \subseteq V, S = \langle t_1, \cdots, t_s \rangle \sqsubseteq T, \Omega \subseteq A \times \{+, -\}\}$$

A *trend dynamic sub-graph* is a pattern of $\mathscr{L}$ whose vertices follow the same trend over Ω. That is to say, the vertex attribute value derivative at a time step t has the same sign over all the vertices and the time steps of the dynamic sub-graph. Many trend dynamic sub-graph can be observed over a dynamic attributed graph, but those that are particularly important occur in vertices that are closely related through the induced sub-graph topology. To that end, we are looking for trend dynamic sub-graphs whose static induce sub-graphs have a small diameter. To summarize, a trend dynamic sub-graph is defined as follows:

> **Definition 3.6 — Trend dynamic sub-graph.** A trend dynamic sub-graph is an element $(\mathscr{G}(U,S), \Omega)$ of $\mathscr{L}$ that satisfies the following properties :
> 1. At each time step $t \in S$, the diameter of the graph $G_t(U)$ is less than or equal to k, where k is a user-defined threshold. I.e., for any two vertices $v, w \in U$, there exists a path connecting them whose length is smaller than or equal to k. Formally, let $d_{G_t(U)}(v,w)$ be the shortest path length between the vertices v and w in $G_t(U)$. The diameter of G

is thus defined by

$$diam_{G_t(U)} \equiv \max_{v,w \in U} d_{G_t(U)}(v,w)$$

and the diameter constraint, that is $diam_{G_t(U)} \leq k, \forall t \in S$, is denoted $\mathscr{C}_{diameter}\big(\mathscr{G}(U,S),\Omega\big)$.

2. Each signed attribute $(a,m) \in \Omega$ defined a trend that has to be satisfied by any vertex $u \in U$ at any timestep $t \in S$:

$$\begin{cases} a^+(u,t) \equiv a(u,t) < a(u,t+1), \, if \, m = + \\ a^-(u,t) \equiv a(u,t) > a(u,t+1), \, if \, m = - \end{cases}$$

This constraint is denoted $\mathscr{C}_{trend}\big(\mathscr{G}(U,S),\Omega\big)$.

3. If $\big(\mathscr{G}(U,S),\Omega\big)$ is maximal, then the sets U and Ω, as well as the sequence S cannot be enlarged without invalidating one or more of the above properties. This constraint is denoted $\mathscr{C}_{maximal}\big(\mathscr{G}(U,S),\Omega\big)$.

3.3.2 Constraints on Trend Dynamic Sub-graphs

To further guide the extraction of trend dynamic sub-graphs toward most relevant ones, we propose several interstingness measures that offer the possibility to the end-users to express their needs. An interestingness measure is a function which assigns a value to a pattern according to its quality. Such a measure can easily be used as a constraint by specifying a user-defined threshold that makes possible the selection of patterns having a high or a low value on these measures.

Size measures: As most simple interestingness measures are often the most useful ones, we first consider size measures that characterize a pattern by the number of elements it contains: $\mathscr{C}_{sz\_vertices}\big(\mathscr{G}(U,S),\Omega\big) = |U|$, $\mathscr{C}_{sz\_times}\big(\mathscr{G}(U,S),\Omega\big) = |S|$ and $\mathscr{C}_{sz\_attributes}\big(\mathscr{G}(U,S),\Omega\big) = |\Omega|$. These measures are generally used to constrain patterns to a minimal size.

Volume measure: In some contexts, it can also be useful to combine the three size measures in a single value: $\mathscr{C}_{volume}\big(\mathscr{G}(U,S),\Omega\big) = \frac{|U|}{|V|} \times \frac{|S|}{|T|} \times \frac{|\Omega|}{|A|}$. This measure is also generally used to constrain patterns to a minimal volume.

Measure of vertex specificity: The question that aims to answer this measure is: How similar are the vertices outside the trend dynamic sub-graph to the ones inside it? We want to quantify the average proportion of trends that are satisfied by outside pattern vertices:

$$\mathscr{C}_{vertex\_specificity}\big(\mathscr{G}(U,S),\Omega\big) = \frac{\sum_{w \in V \backslash U} \sum_{(a,m) \in \Omega} \sum_{t \in S} \delta_{a^m(w,t)}}{|V \backslash U| \times |\Omega| \times |S|}$$

where $\delta_{condition}$ is the Kronecker function that is equal to 1 if *condition* is satisfied, or 0 otherwise. The more the trend dynamic sub-graph is made of specific vertices with respect to attribute trends, the lower this measure.

Measure of trend relevancy: The question that aims to answer this measure is: Does the attributes that do not belong to Ω have an homogeneous trend on $\mathscr{G}(U,S)$? To that end, we evaluate the entropy of the attribute trends and consider the one that has the smallest entropy. Let

$$P_1\big(b^m,\mathscr{G}(U,S)\big) = \frac{\sum_{u \in U} \sum_{t \in S} \delta_{b^m(u,t)}}{\sum_{u \in U} \sum_{t \in S} \big(\delta_{b^-(u,t)} + \delta_{b^+(u,t)}\big)}$$

be the proportion of the trend m of attribute b on the vertices and time steps of $\mathscr{G}(U,S)$. Then the trend relevancy interestingness measure is:

$$\mathscr{C}_{trend\_relevancy}\big(\mathscr{G}(U,S),\Omega\big) = \min_{b \in A\setminus\Omega} \sum_{m\in\{-,+\}} -P_1(b^m,\mathscr{G}(U,S))\log P_1(b^m,\mathscr{G}(U,S))$$

The more a trend dynamic sub-graph is trend relevant, the higher this measure.

Measure of temporal dynamic: The question that aims to answer this measure is: How does a pattern appear in the time? Does it burst? To that end, we evaluate the dynamic of the proportion of vertices and attributes that satisfy the pattern before and after the time steps of S: $P_2(t,(\mathscr{G}(U,S),\Omega)) = \frac{\sum_{u\in U}\sum_{(a,m)\in\Omega}\delta_{a^m}(u,t)}{|U|.|\Omega|}$. If a trend dynamic sub-graph bursts, then the proportion P_2 is below a threshold at every time steps not in S:
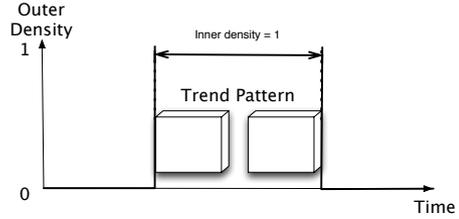
$$\mathscr{C}_{temporal\_dynamic}(\mathscr{G}(U,S),\Omega) = \max_{t\in T\setminus S} P_2(t,(\mathscr{G}(U,S),\Omega))$$

3.3.3 Constraint properties

We consider the following partial order between attributed induced dynamic sub-graphs.

> **Definition 3.7 — Partial order on attributed induced dynamic sub-graphs.** Let $Q_1 = \big(\mathscr{G}(U_1,S_1),\Omega_1\big)$ and $Q_2 = \big(\mathscr{G}(U_2,S_2),\Omega_2\big)$ be two attributed induced dynamic sub-graphs. We say that Q_1 is more specific than Q_2, $Q_1 \preceq Q_2$, iff $U_1 \subseteq U_2$ and $S_1 \subseteq S_2$ and $\Omega_1 \subseteq \Omega_2$.

Considering this partial order, we propose to compute attributed induced dynamic sub-graphs using the generic algorithm presented in Section 1.4. Let us just recall the notations used in this algorithm, before considering the constraints properties and the way they are handle in this algorithm. $\mathscr{R} = \{Q_i \mid i = 1\ldots k\}$ denotes a nonempty finite set of attributed induced dynamic sub-graphs. $\mathscr{R}^\vee = \big(\mathscr{G}(\bigcup U_i,\bigcup S_i),\bigcup\Omega_i\big)$ and $\mathscr{R}^\wedge = \big(\mathscr{G}(\bigcap U_i,\bigcap S_i),\bigcap\Omega_i\big)$ are respectively the join and meet elements of $\mathscr{R}$.

Trend constraint: This constraint is anti-monotone with respect to $\preceq$. That is, if Q_1 and Q_2 are two attributed induced dynamic sub-graphs such that $Q_1 \preceq Q_2$, then $\mathscr{C}_{trend}(Q_2) \Rightarrow \mathscr{C}_{trend}(Q_1)$. The anti-monotone property of the *trend* constraint implies that if $\mathscr{C}_{trend}(\mathscr{R}^\wedge)$ is not satisfied, then $\mathscr{R}$ is empty. This constraint can also be propagated using the following procedure: if there exists e in $\mathscr{R}^\vee \setminus \mathscr{R}^\wedge$ such that $\mathscr{C}_{trend}(\mathscr{R}^\wedge \cup e)$ is not satisfied, then e is removed from $\mathscr{R}^\vee$.

Diameter constraint: This constraint is neither monotone nor anti-monotone with respect to $\preceq$. However, noting that this constraint is monotone or anti-monotone in each of its parameters, we can derive a propagation mechanism of this constraint. That is, for all vertex v and all time step t in the trend sub-graph, we should have $\max_{w\in U_1} d_{G_t(U_2)}(v,w) \leq k$. This constraint is anti-monotone on U_1 and monotone on U_2, that is (a) if the constraint is satisfied on U_1, it is also satisfied for any of its subsets; (b) if the constraint is satisfied on a graph $G_t(U_2)$, then, adding some vertices and edges to $G_t(U_2)$ will not increase its value. Therefore, in MINTAG algorithm, this constraint is propagated using the following mechanisms: (1) if there exists $v \in \mathscr{R}^\vee \setminus \mathscr{R}^\wedge$, $w \in \mathscr{R}^\wedge$ and $t \in \mathscr{R}^\wedge$ such that $d_{G_t(\mathscr{R}^\vee \cup V)}(v,w) > k$ then v is removed from $\mathscr{R}^\vee$; (2) if there exists $t \in \mathscr{R}^\vee \setminus \mathscr{R}^\wedge$, $v \in \mathscr{R}^\wedge$ and $w \in \mathscr{R}^\wedge$ such that $d_{G_t(\mathscr{R}^\vee \cup V)}(v,w) > k$ then t is removed from $\mathscr{R}^\vee$.

Minimal size constraints: These constraints are monotone. Thus, if $\mathscr{C}_{sz\_vertices}(\mathscr{R}^\vee \cap V) <$ min_sz_vertices or $\mathscr{C}_{sz\_attributes}(\mathscr{R}^\vee \cap A \times \{+,-\}) <$ min_sz_attributes or $\mathscr{C}_{sz\_times}(\mathscr{R}^\vee \cap T) <$ min_sz_times, then $\mathscr{R}$ is empty.

Minimal volume constraint: Similarly, this constraint is monotone and if $\mathscr{C}_{volume}(\mathscr{R}^\vee) <$ min_volume, then $\mathscr{R}$ is empty.

Maximal vertex_specificity constraint: As the diameter constraint, this constraint is monotone or anti-monotone on each of its parameters. Considering the equation $\frac{\sum_{w \in V \setminus U_1} \sum_{(a,m) \in \Omega_1} \sum_{t \in S_1} \delta_{a^m(w,t)}}{|V \setminus U_2| \times |\Omega_2| \times |S_2|} \leq$ max_vertex_spec, we can observe that it is monotone on U_1, S_2 and Ω_2 and anti-monotone on U_2, S_1 and Ω_1. Thus, $\mathscr{R}$ is empty if

$$\frac{\sum_{w \in (\mathscr{R}^\vee \cap V)} \sum_{(a,m) \in (\mathscr{R}^\wedge \cap (A \times \{+,-\}))} \sum_{t \in (\mathscr{R}^\wedge \cap T)} \delta_{a^m(w,t)}}{|\mathscr{R}^\wedge \cap V| \times |\mathscr{R}^\vee \cap (A \times \{+,-\})| \times |\mathscr{R}^\vee \cap T|} > \text{max\_vertex\_spec}$$

Minimal trend_relevancy constraint: Handling this constraint is a little more tricky. Let us first consider the entropy function with two probability values: $f(x) = -x \log(x) - (1-x) \log(1-x)$. This function increases on $[0, \frac{1}{2}]$ and decreases on $[\frac{1}{2}, 1]$. Using this notation, the minimal trend_relevancy can be rewritten as $\min_{b \in A \setminus \Omega} f(P_1(b^+, \mathscr{G}(U,S))) \geq$ min_trend_rel.[4] Second, we can derive the following upper bound on $P_1(b^m, \mathscr{G}(U,S))$:

$$P_1(b^m, \mathscr{G}(U,S)) \leq \frac{\sum_{u \in (\mathscr{R}^\vee \cap U)} \sum_{t \in (\mathscr{R}^\vee \cap S)} \delta_{b^m(u,t)}}{\sum_{u \in (\mathscr{R}^\wedge \cap U)} \sum_{t \in (\mathscr{R}^\wedge \cap S)} \left(\delta_{b^-(u,t)} + \delta_{b^+(u,t)} \right)} = UB(b^m)$$

as P_1 is monotone on its numerator parameters, and anti-monotone on its denominator ones. Similarly, we can derive a lower bound[5] $LB(b^m) \leq P_1(b^m, \mathscr{G}(U,S))$. Thus, if $UB(b^m) \leq \frac{1}{2}$, then f is increasing and $f(P_1(b^m, \mathscr{G}(U,S))) \leq f(UB(b^m))$. Similarly, if $LB(b^m) \geq \frac{1}{2}$, then f is decreasing and $f(P_1(b^m, \mathscr{G}(U,S))) \leq f(LB(b^m))$.

Therefore, if there exists $b \in A \setminus \mathscr{R}^\vee$ and $m \in \{+,-\}$ such that either (1) $UB(b^m) \leq \frac{1}{2}$ and $f(UB(b^m)) <$ min_trend_rel, or (2) $LB(b^m) \geq \frac{1}{2}$ and $f(LB(b^m)) <$ min_trend_rel then $f(P_1(b^m, \mathscr{G}(U,S))) <$ min_trend_rel and we can conclude that $\mathscr{R}$ is empty.

Maximal temporal_dynamic constraint: This constraint is anti-monotone on its parameters on the numerator and monotone on the ones on the denominator:

$$\max_{t \in T \setminus S} \frac{\sum_{u \in U} \sum_{(a,m) \in \Omega} \delta_{a^m(u,t)}}{|U|.|\Omega|} \leq \text{max\_temp\_dyn}$$

Therefore, if there exists $t \in T \setminus \mathscr{R}^\vee$ such that $\frac{\sum_{u \in \mathscr{R}^\wedge \cap U} \sum_{(a,m) \in \mathscr{R}^\wedge \cap \Omega} \delta_{a^m(u,t)}}{|\mathscr{R}^\vee \cap U|.|\mathscr{R}^\vee \cap \Omega|} > \text{max\_temp\_dyn}$, then we can conclude that $\mathscr{R}$ is empty.

3.3.4 Experimental Study

We considered three real-world dynamic attributed graphs whose characteristics are given in Table 3.8.

---

[4]This is equivalent to $\min_{b \in A \setminus \Omega} f(P_1(b^-, \mathscr{G}(U,S))) \geq$ min_trend_rel as $P_1(b^+, \mathscr{G}(U,S)) = 1 - P_1(b^-, \mathscr{G}(U,S))$.

[5] $LB(b^m) = \frac{\sum_{u \in (\mathscr{R}^\wedge \cap U)} \sum_{t \in (\mathscr{R}^\wedge \cap S)} \delta_{b^m(u,t)}}{\sum_{u \in \mathscr{R}^\vee \cap U} \sum_{t \in (\mathscr{R}^\vee \cap S)} \left(\delta_{b^-(u,t)} + \delta_{b^+(u,t)} \right)} \leq P_1(b^m, \mathscr{G}(U,S))$

Dynamic attributed graph		$|V|$	$|T|$	$|A|$	density
DBLP		2145	10	43	1.3×10^{-3}
US Flights	Last 20 years	361	20	8	3.2×10^{-2}
	September 2001	220	30	6	5.7×10^{-2}
	Two years around 9/11	234	25	8	5.7×10^{-2}
	Katrina	280	8	8	5×10^{-2}
Brazil landslides		394885	2	11	5.7×10^{-4}

Table 3.8: Main characteristics of the dynamic attributed graphs.

DBLP: This co-authorship graph is built from the DBLP digital library [6]. Each vertex represents an author who published at least ten papers in one of the major conferences and journals of the Data Mining and Database communities between January 1990 and December 2012. This time period is divided in 10 time steps. Each time step describes the co-authorship relations and the publication records of the authors over 5 consecutive years. For sake of consistency in the data, two consecutive periods have a 3 year overlap[7]. Each edge at a time step t links two authors who co-authored at least one paper in this time interval. The vertex properties are the number of publications in each of the 43 journals or conferences.

US Flights: RITA "On-Time Performance" database[8] contains on-time arrival data for non-stop US domestic flights by major air carriers. From this database, we generated 4 dynamic attributed graphs that aggregate data over different period of time. Graph vertices stand for US airports and are connected by an edge if there is at least a flight connecting them during the time period. We consider 8 vertex attributes that are the number of departures/arrivals, the number of canceled flights, the number of flights whose destination airport has been diverted, the mean delay of departure/arrival and the ground waiting time departure/arrival. The four dynamic graphs are:

- Last 20 years: Data are aggregated over each year.
- September 2001: Data are aggregated over each day of September 2001.
- Two years around 9/11: Data are aggregated over each month between September 2000 and September 2002.
- Katrina: To study the consequences of hurricane Katrina on US airports, data are aggregated over each week between 01/08/2005 and 25/09/2005.

Brazil landslides: This dynamic attributed graph is derived from two satellite images taken before and after huge landslides in Brazil. It is composed of 394885 vertices that stand for image shapes (segmented areas), two time steps and 11 attributes that are the spectral response in infra-red, red, blue green and indices computed from these values. There is an edge between two vertices if the corresponding shapes are contiguous.

The ensuing experimental study aims at answering the following questions: *What is the efficiency of the algorithm with regard to the graph characteristics that may affect its execution time? How effective are the pruning properties? What about the trend dynamic sub-graph relevancy?*
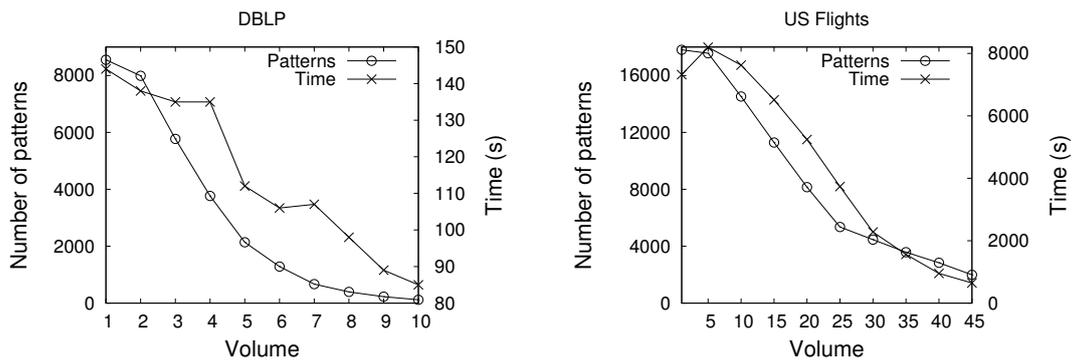
Figure 3.13: Number of patterns and runtime for DBLP (left) and US flights (right) with respect to volume: max_vertex_spec = 0.5, min_trend_rel = 0.05 and max_temp_dyn = 0.8. The diameter is set to 2 on (left) and to 1 on (right).

Quantitative Results

Figure 3.13 shows the number of extracted patterns and the execution times of the algorithm on DBLP and US Flights with respect to the volume threshold. When the minimum volume threshold decreases, more execution time is required since more trend dynamic sub-graph are obtained. Yet, the algorithm is able to extract trend dynamic sub-graphs when the minimum volume threshold is minimal, that is to say equals 1, since we report absolute volume values. It does not exhibit a similar monotonic behavior when varying the diameter constraint: the time computation is no more proportional to the number of extracted patterns. Actually, pushing this constraint needs to compute shortest paths in the graph, that is costly.

Figure 3.14 reports the execution times and the number of patterns with respect to the other constraints: $\mathscr{C}_{vertex\_specificity}$, $\mathscr{C}_{trend\_relevancy}$ and $\mathscr{C}_{temporal\_dynamic}$. We can observe that for the graphs DBLP and US Flights, the less stringent the constraints, the higher the execution times and the number of patterns are. In most of the cases, the number of patterns increases dramatically. This behavior shows that our approach push efficiently these constraints that are neither monotone nor anti-monotone. It is noteworthy that in Figure 3.14, the execution time on DBLP for min_trend_rel = 0 is not available because the process was killed after several hours.

Figure 3.15 reports on the scability of the algorithm. We used DBLP and replicated alternatively the number of vertices, time steps and attributes. As the number of extracted patterns is not preserved by these replications (i.e., the vertex replication adds connected components while the time replication introduces new variations involving the last time step) we report the runtime per pattern. It appears that the algorithm is more robust to the increase of the number of attributes and to the number of vertices than to the number of time steps. This is a good point since, in practice, the number of vertices is often large while the numbers of attributes and the number of time steps are rather small.

We study the effectiveness of each constraint on both DBLP and US Flights, when varying the different thresholds (volume, vertex specificity, temporal dynamic and trend relevancy). To this end, we count the number of pruned unpromising candidates by each constraint. The results are shown in Figure 3.16 for DBLP (top) and US Flights (bottom). It is noteworthy that all the constraints enable to prune unpromising candidates and they have different impact on both graphs. We can observe that the trend relevancy constraint is effective on the two graphs and prunes almost 50% of the unpromising candidates on DBLP in most of the cases. Even if this

---

[6]http://dblp.uni-trier.de/
[7][1990-1994][1992-1996][1994-1998]...[2008-2012]
[8]http://www.transtats.bts.gov
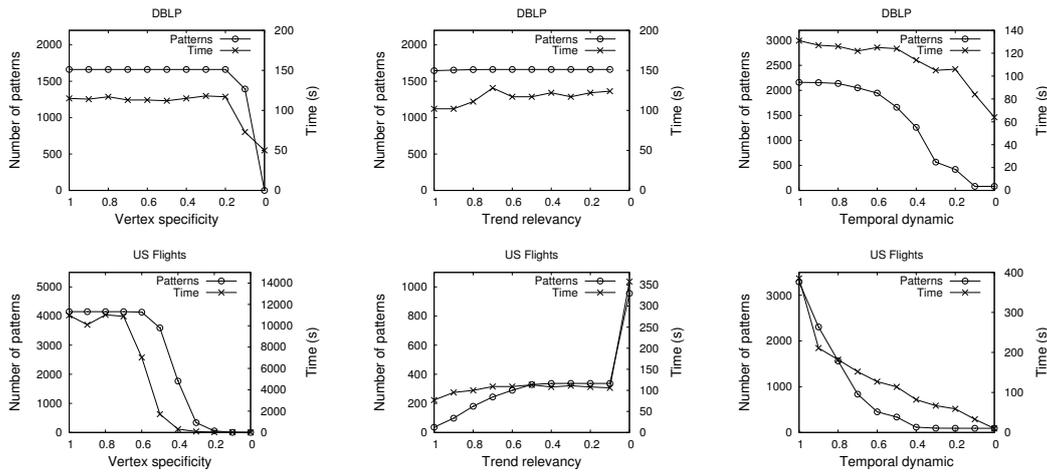
Figure 3.14: Runtime and number of patterns with respect to the specificity measures (max_vertex_spec = 0.3, min_trend_rel = 0.1, max_temp_dyn = 0.5, min_volume = 5 and max_diameter = 2 for DBLP (top) or 1 for US flights (bottom)).
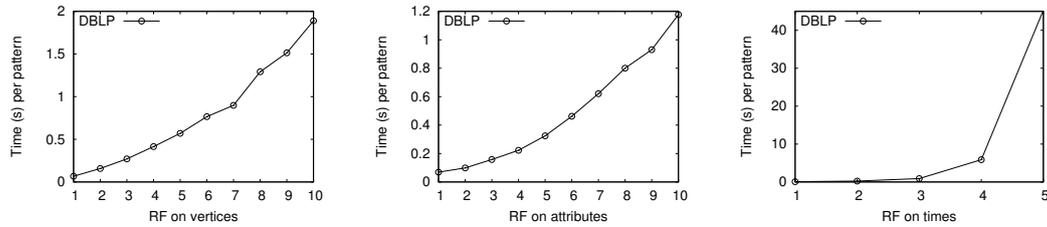
Figure 3.15: Runtime per pattern with respect to replication factors on vertices, attributes and time steps (max_vertex_spec = 0.3, min_trend_rel = 0.1, max_temp_dyn = 0.5, min_volume = 5 and max_diameter = 2.

constraint has no anti-monotone property, it is efficiently pushed. The volume constraint, more effective on DBLP than US Flights, makes possible to prune large part of the search space. This behavior is much more expected since this constraint is anti-monotone. The pruning impact of the temporal_dynamic constraint is not negligible, since it prunes nearly 20% of the candidates on DBLP and up to 60% on US flights. This important difference is mainly due to the temporal regularity of US Flights. This can also explain the fact that the vertex specificity constraint plays a prominent role on the US Flights while having a limited impact of the DBLP dynamic graph.

Qualitative Results

Results on DBLP: We perform an extraction on DBLP dynamic attributed graph with parameter max_diameter set to infinity (vertices belong to the same connected component) and min_volume = 5. Other constraints threshold are set so as not to constrain the result. We obtained 112 trend dynamic sub-graphs in less than 4 seconds. The top 2 largest patterns depict the same well-known phenomenon, explained below. The first pattern involves 171 authors having an increasing number of publications in PVLDB between 2004 and 2012. The second one involves 164 authors that have a decreasing number of publications in VLDB during the same period. These patterns reflect the new policy of the VLDB endowment. Indeed, PVLDB appeared in 2008 and, in 2010, the review process of the VLDB conference series was done in collaboration with, and entirely through PVLDB in 2011. Then, we carry out a new extraction taking into account all the constraints (max_diameter = 2, max_vertex_spec = 0.3, max_temp_dyn = 0.5)
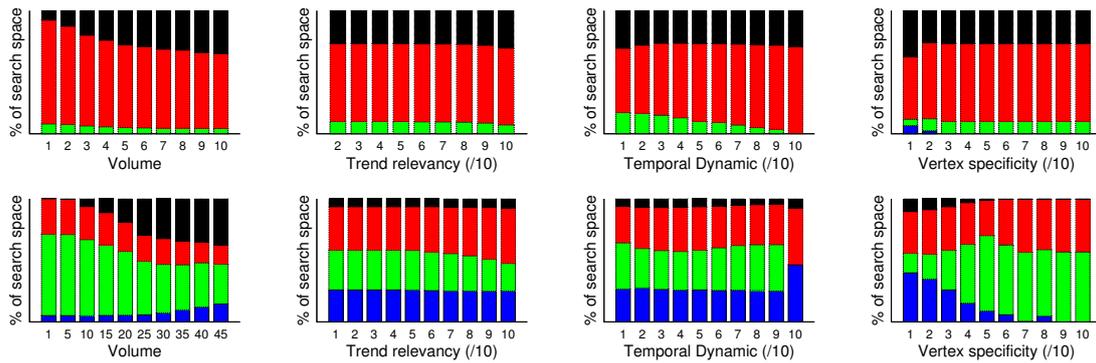
Figure 3.16: Constraint efficiency on DBLP (top) and US Flights (bottom) w.r.t. specificity measures. From top to bottom: volume (black), trend_relevancy (red), temporal_dynamic (green) and vertex_specificity (blue). Same parameters as in Figures 3.13 and 3.14.

except min_trend_rel that was set to 0. We obtained 41 patterns in 8 seconds. We first consider the pattern that has the longest duration and involves the most recent period, that is [2008-2012]. It implies the vertices related to Jimeng Sun and Christos Faloutsos, who have an increasing number of publications in KDD and SDM, while having a decreasing number of publications in VLDB. We consider another pattern which has the best *temporal_dynamic* value among the patterns having their *trend_relevancy* greater than 0.1. It involves two authors, Rong Zhou and Eric A. Hansen, and the time steps between 1998 and 2008. On this period, the authors have an increasing number of publications in AAAI conference series. This pattern has good values on *vertex_specificity* (0.12), *temporal_dynamic* (0) and *trend_relevancy* (0.81). This publication trend is rare with regard to the whole graph.

Figure 3.17: Airports (left) involved in the top temporal_dynamic trend dynamic sub-graph (in red) and in the top trend_relevancy (in yellow) and the Katrina's track (right).

Results on Katrina: Hurricane Katrina was the deadliest and most destructive Atlantic hurricane of the 2005 Atlantic hurricane season. It was the costliest natural disaster, as well as one of the five deadliest hurricanes, in the history of the United States. Among recorded Atlantic hurricanes, it was the sixth strongest overall. In this experiment, we aim to characterize the impact of this hurricane on the US domestic flights. To this end, we set constraints as follows: min_volume = 10, max_vertex_spec = 0.6, min_trend_rel = 0.1, max_temp_dyn = 0.2 and max_diameter = ∞. We extract 37 patterns in 14 seconds. We look for two patterns: (i) the trend dynamic sub-graph with largest *temporal_dynamic* value, and (ii) the pattern with the highest *trend_relevancy* value. These patterns and Katrina's track[9] are shown in Figure 3.17. Pattern (i) involves 71 airports (in red on Figure 3.17 (left)) whose arrival delays increase over 3 weeks.

---

[9]Map from ©2013 Google, INEGI, Inav/Geosistemas SRL, MapLink
http://commons.wikimedia.org/wiki/File:Katrina_2005_track.png

Pattern	$|V|$	Days	A	*vertex_spec.*	*temp._dyn.*	*trend_rel.*
P1	179	10, 11	#Cancel.$^+$	0.5	0.41	0.94
P2	111	13, 15	#Cancel.$^-$	0.52	0.83	0.9
P3	102	13, 14, 15	#Cancel.$^-$	0.6	0.84	0.81

Table 3.9: Trend dynamic sub-graphs extracted on September 2001 graph.

One week is not related to the hurricane but the two others are the two weeks after Katrina caused severe destruction along the Gulf coast. This pattern has a *temporal_dynamic* $= 0$, which means that arrival delays never increased in these airports during another week. The hurricane strongly influenced the domestic flight organization. Pattern (ii) has a *trend_relevancy* value equal to 0.81 and includes 5 airports (in yellow on Figure 3.17 (left)) whose number of departures and arrivals increased over the three weeks following Katrina hurricane. Three out of the 5 airports are in the Katrina's trajectory while the two other ones were impacted because of their connections to airports from damaged areas. Substitutions flights were provided from these airports during this period. The values on the other interestingness measures show that this behavior is rather rare in the rest of the graph (*vertex_specificity* $= 0.29$, *temporal_dynamic* $= 0.2$).

Results on September 2001: To characterize the impact of September 11 attacks, we look for patterns involving many airports (at least 100) whose trends are relevant (*trend_relevancy* $= 0.8$). Given this setting, the algorithm returns 3 trend dynamic sub-graphs in 8 seconds. These patterns are reported in Table 3.9. They depict a large number of airports, whose number of canceled flights increased on September 11 and 12 compared to the previous days, and then decreased two days after the terrorist attacks (between the 13th and 16th September). These patterns identify the time required for a return to normal domestic traffic.

Results on Two years around 9/11: Considering longer periods before and after the September attacks, with more restrictive threshold values (*temporal_ dynamic* $= 1$, *vertex_specificity* $= 0.5$ and *trend_relevancy* $= 0.8$), we obtain 87 patterns in 67 seconds. The top *trend_relevancy* pattern involves 159 airports that have an increasing number of canceled flights in September 2001 and December 2000. Obviously, the number of canceled flights in September 2001 is related to terrorist attack. It is noteworthy that December 2000 snow storm had a similar impact on the cancellation of flights, because we do not quantify the strength of the trends. Actually, the number of canceled flights in September 2001 is four times bigger than the one in December 2000.

Results on Brazil landslides: In this series of 2 satellite images, the goal is to identify regions in which a landslide appears in the second image. Generally, the main consequence of a landslide if the disappearance of the vegetation. Therefore, we focus on the patterns that involve $NDVI^-$, since $NDVI$ is a computed index that quantifies the level of vegetation. The algorithm returns 4821 patterns in 2 hours that involve 34275 regions that are reported on Figure 3.18. These results were evaluated by an expert who testified that 69% of the true landslide regions appear in the computed patterns. These regions represent 46% of the extracted regions. The 54% remaining regions belong to one of the 4 following categories:(1) regions nearby true landslides which have not been interpreted as landslides by the expert (border effect), (2) deforested area not due to landslides (e.g., human activity), (3) regions found due to misalignment of the segmentation technique and (4) regions that represent cities and human activity footprints.
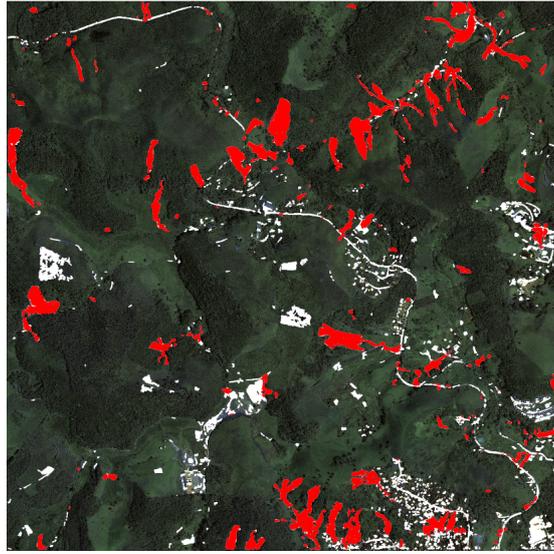
Figure 3.18: Regions involved in the patterns: true landslides (red) and other phenomena (white).

3.4 Discussion

In this chapter, we showed several examples of constraint-based pattern mining frameworks for attributed and/or dynamic relational graphs. One of the difficulties when analyzing attributed relational graphs is to design patterns that characterize both the structure encoded by the graph relationship and the attribute values of the vertices. Our first proposal was to analyze static attributed graphs using a propositionalization of the graph structure based on topological measures evaluated at each vertex. Topological patterns, that mix topological measures and attribute values, are extracted to find co-variations between some descriptors of vertex connectivity and vertex attribute values. Our second contribution concerns the discovery of patterns in relational dynamic graphs. We proposed a local to global approach that studies time-evolving patterns: It first extracts highly connected sub-graphs, and then looks for relationships between patterns of two consecutive time steps. Our third proposal aims to extract patterns in relational attributed and dynamic graphs. We introduced a new approach that mines constrained dynamic sub-graphs: The sub-graph diameter is constrained so that to limit the length of the longest shortest path between any two vertices of the sub-graph (constraints such as pseudo-cliques are too stringent and tend to fragment some reliable patterns) and the attributes associated to the dynamic sub-graph follow the same trend over the time steps associated to the pattern.

The prospects of this work are many, as attributed dynamic graph data are numerous and require the design of suitable pattern domains that depends on the application domains. To cite just a few, attribute values may be embedded into a hierarchy structure that gives the possibility to generalize or specialize the patterns; Or one can be interested in triggering attributes whose changes impact the graph topology; Finally, the dynamics of the edges can be the center of the study and we may extract temporal dependencies between edge activation.
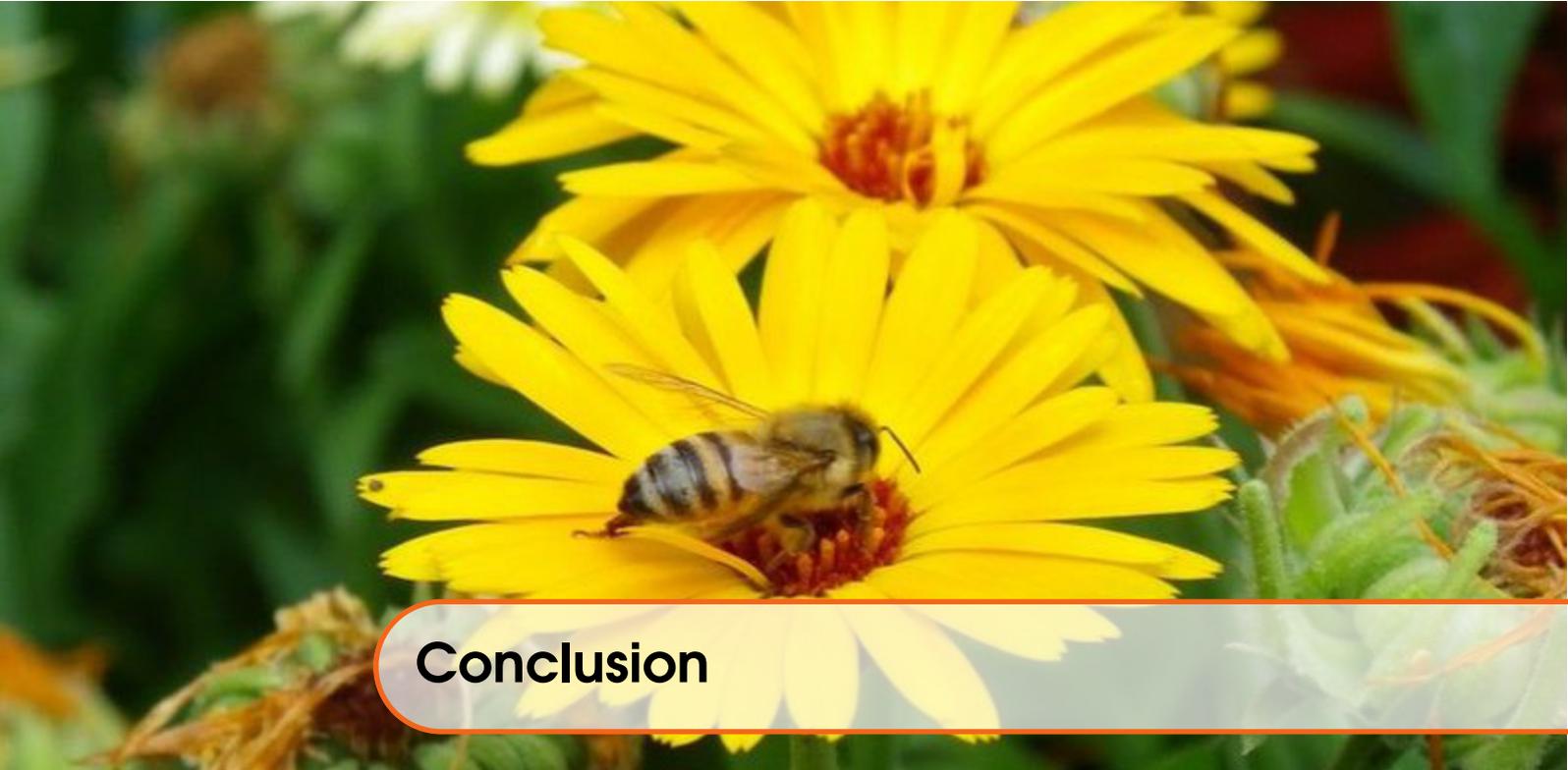
Conclusion

In this *Habilitation à Diriger des Recherches* thesis, I present the main results I have contributed to in the areas of local pattern extraction under constraints and in the analysis of dynamic graphs. In the coming years, our research will be guided by two main projects that have just started or been accepted. The first one, named Vél'innov [10], is a continuation of the work presented in Section 2.2. From the records of all movements of the Vélo'v bicycle sharing system since its creation, as well as information about subscribers obtained by quantitative and qualitative user surveys, this research project aims to:

- Characterize the bicycle sharing system as an evolving socio-technical system in order to be able to model and to simulate the impacts of local changes in the city or in the system;
- Analyze how the persons or classes of persons tame this socio-technical innovation, as well as the impact on practices and representations of the system;
- Develop a set of statistical tools and methods based on the analysis of this particular complex systems and on interdisciplinary collaboration among project partners.

In this context, we will continue our exploration of dynamic attributed graphs. A fist direction will be to seek for relationships between attribute value changes and topological modifications of the graph. We plan to discover temporal combinations of attribute changes that are strongly correlated with significant topological modifications within the dynamic graph. A second direction will be to use vertex attributes to select a part of the dynamic graph, transform it into signals, and then analyze these signals using signal processing methods. We propose to use classical multidimensional scaling (CMDS) to transform the network into time series. This method generates signals which preserves the adjacency matrix of the graph viewed as dissimilarities between the vertices (Hamon, Borgnat, Flandrin, and Robardet 2013a; Hamon, Borgnat, Flandrin, and Robardet 2013b). Signals will then be characterized by either a frequency analysis, or by the search of discriminating patterns.

The second project, named GRAISearch (Use of Graphics Rendering and Artificial Intelligence for Improved Mobile Search Capabilities) [11], involves three partners: (1) Tapastreet Ltd is an Irish company that focuses on the development and commercialization of a location based social media search engine platform that, in its current form, returns geo-located video and image media from major social networks for any location and any topic anywhere in the

---

world; (2) Trinity College Dublin (TCD) – represented by the group of Professor Rozenn Dahyot from the Computer science and statistics department / Graphic Vision and Visualisation Centre (GV2) – focuses on the development of Video summarization techniques and 3D Scene rendering; (3) INSA-Lyon – represented by the DM2L LIRIS team (Data Mining & Machine Learning), that works on knowledge discovery in databases. In this project, we first plan to work on the development of a trajectory mining algorithm to predict local demographic flows using geo-tagged social media uploaded data. The growth of location based smart-phone applications makes possible to collect significant quantities of user data that can be use to make local demographic flow predictions. The second contribution would be related to the proposition of a local event detection algorithm to identify local breaking news events. Similar approaches have been proposed to detect global events (earthquakes, pandemics) from social media data, but none as yet have been designed for a vast quantity of useful local newsworthy events. A major benefit of developing a new automatic event detection system from uploaded geo-located photos, videos, social preferences (Likes, Shares etc) is the added social feedback component: We aim to assess the impact an event had and how people reacted to it. This will be investigated by modelling the problem of trust sources identification as an original problem of temporal dependency discovery between topics from different social media that will be the support to intelligently detect breaking news with respect to the context. It is very important to take benefit from the context such as the geo-localization and the social characteristics of the individuals. A deep analysis of the propagation of the information in social networks will enable to both suggest targets to crawl and assess the results of the social media search engine. Indeed, identifying the domain leaders is important to rank the results and to automatically compute a reputation score on new comers. Finally, those techniques will be integrated into a geo-located social media recommender system.

These projects give us original application contexts for which we will design pattern domains that support appropriate knowledge discovery and continue our theoretical studies on generic mechanisms that sustains the constraint-based pattern mining framework.

Before completing this manuscript, I would like to explain the choice of photos that illustrate each chapter: The product of a fruitful fishing represents the patterns obtained by a successful data mining process; The bicycle shape candy expresses the fun I have to work with the team of *Vélo'ver researchers*; And the colorful fireworks symbolize the attributed dynamic graphs and the recreation that their study provides me; Without forgetting the little bee that is all that I have to discover.

Bibliography

Abarbanel, Henry (1996). *Analysis of Observed Chaotic Data*. Springer (cited on page 37).

Agrawal, Rakesh and Ramakrishnan Srikant (1994). "Fast Algorithms for Mining Association Rules in Large Databases". In: *Proceedings of the 20th International Conference on Very Large Data Bases*. VLDB '94. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., pages 487–499. ISBN: 1-55860-153-8 (cited on page 21).

— (1995). *Mining Sequential Patterns*. Technical report. Almaden Reasearch Center, 650 Harry Road, San Jose, CA 95120-6099: IBM Research Division (cited on page 24).

Albert, Réka and Albert-László Barabási (2000). "Topology of evolving networks: local events and universality". In: *Physical review letters* 85.24, pages 5234–5237 (cited on page 60).

Avis, David and Komei Fukuda (1996). "Reverse Search for Enumeration". In: *Discrete Applied Mathematics* 65.1-3, pages 21–46 (cited on page 13).

Barabási, Albert-László and Réka Albert (1999). "Emergence of scaling in random networks". In: *Science* 286.5439, pages 509–512 (cited on page 35).

Basseville, Michèle (1989). "Distance measures for signal processing and pattern recognition". In: *Signal Processing* 18.4, pages 349–369 (cited on page 55).

Besson, Jérémy (2005). "Découvertes de motifs pertinents pour l'analyse du transcriptome : application à l'insulino-résistance". Ecole Doctorale Informatique et Information pour la Société. PhD thesis. INSA de Lyon (cited on page 17).

Besson, Jérémy, Céline Robardet, and Jean-François Boulicaut (2004). "Constraint-Based Mining of Formal Concepts in Transactional Data". In: *Advances in Knowledge Discovery and Data Mining, 8th Pacific-Asia Conference, PAKDD 2004, Sydney, Australia, May 26-28, 2004, Proceedings*. Edited by Honghua Dai, Ramakrishnan Srikant, and Chengqi Zhang. Volume 3056. Lecture Notes in Computer Science. Springer, pages 615–624 (cited on pages 8, 15).

— (2006). "Mining a New Fault-Tolerant Pattern Type as an Alternative to Formal Concept Discovery". In: *Conceptual Structures: Inspiration and Application, 14th International Conference on Conceptual Structures, ICCS 2006, Aalborg, Denmark, July 16-21, 2006, Proceedings*. Edited by Henrik Schärfe, Pascal Hitzler, and Peter Øhrstrøm. Volume 4068. Lecture Notes in Computer Science. Springer, pages 144–157 (cited on page 8).

Besson, Jérémy, Céline Robardet, Jean-François Boulicaut, and Sophie Rome (2005). "Constraint-based Formal Concept Mining and its Application to Microarray Data Analysis". In: *Intelligent Data Analysis* 9.1, pages 59–82 (cited on pages 8, 15, 17, 35).

Blachon, Sylvain, Ruggero Pensa, Jérémy Besson, Céline Robardet, Jean-François Boulicaut, and Olivier Gandrillon (2007). "Clustering formal concepts to discover biologically relevant knowledge from gene expression data." In: *In Silico Biology* 7.0033 (cited on page 35).

Blockeel, Hendrik, Toon Calders, Élisa Fromont, Bart Goethals, Adriana Prado, and Céline Robardet (2008). "An inductive database prototype based on virtual mining views". In: *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Las Vegas, Nevada, USA, August 24-27, 2008*. Edited by Ying Li, Bing Liu, and Sunita Sarawagi. ACM, pages 1061–1064 (cited on page 9).

— (2012). "An inductive database system based on virtual mining views". In: *Data Min. Knowl. Discov.* 24.1, pages 247–287 (cited on page 8).

Blondel, Vincent, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre (2008). "Fast unfolding of communities in large networks". In: *Journal of Statistical Mechanics: Theory and Experiment* P10008.10 (cited on page 51).

Boccaletti, Stefano, Vito Latora, Yamir Moreno, Martin Chavez, and D-U Hwang (2006). "Complex Networks : Structure and Dynamics". In: *Physics Reports* 424.4-5, pages 175–308 (cited on page 35).

Boden, Brigitte, Stephan Günnemann, and Thomas Seidl (2012). "Tracing clusters in evolving graphs with node attributes". In: *CIKM*, pages 2331–2334 (cited on page 82).

Bodon, Ferenc (2005). "A trie-based APRIORI implementation for mining frequent item sequences". In: *OSDM '05: Proceedings of the 1st International Workshop on Open Source Data Mining*. Chicago, Illinois: ACM, pages 56–65. ISBN: 1-59593-210-0 (cited on page 78).

Bonchi, Francesco, Fosca Giannotti, Alessio Mazzanti, and Dino Pedreschi (2005). "Exante: A Preprocessing Method for Frequent-Pattern Mining". In: *IEEE Intelligent Systems* 20.3, pages 25–31 (cited on page 33).

Bonchi, Francesco and Claudio Lucchese (2007). "Extending the state-of-the-art of constraint-based pattern discovery". In: *Data & Knowledge Engineering* 60.2, pages 377–399 (cited on page 13).

Borgnat, Pierre, Patrice Abry, Patrick Flandrin, Céline Robardet, Jean-Baptiste Rouquier, and Eric Fleury (2011). "Shared Bicycles in a City: A Signal processing and Data Analysis Perspective". In: *Advances in Complex Systems,* 14.3, pages 1–24 (cited on pages 9, 48–50).

Borgnat, Pierre, Patrice Abry, Patrick Flandrin, and Jean-Baptiste Rouquier (2009). "Studying Lyon's Velo'V: A Statistical Cyclic Model". In: *Proceedings of European Conference on Complex Systems (ECCS'09)* (cited on pages 48, 50).

Borgnat, Pierre, Eric Fleury, Céline Robardet, and Antoine Scherrer (2009). "Spatial analysis of dynamic movements of Vélo'v, Lyon's shared bicycle program". In: *Proceedings of European Conference on Complex Systems (ECCS'09)*. Edited by François Kepes (cited on page 50).

Borgnat, Pierre, Céline Robardet, Patrice Abry, Patrick Flandrin, Jean-Baptiste Rouquier, and Nicolas Tremblay (2013). "A Dynamical Network View of Lyon's Vélo'v Shared Bicycle System". In: *Dynamics of Time-Varying Networks*. Edited by N. Ganguly, A. Mukherjee, B. Mitra, F. Peruani, and M. Choudhury. Springer (cited on pages 9, 36, 48).

Borgwardt, Karsten M., Hans-Peter Kriegel, and Peter Wackersreuther (2006). "Pattern mining in frequent dynamic subgraphs". In: *ICDM*. IEEE, pages 818–822 (cited on page 75).

Böttcher, Mirko, Frank Höppner, and Myra Spiliopoulou (2008). "On exploiting the power of time in data mining". In: *SIGKDD Explor. Newsl.* 10.2, pages 3–11. ISSN: 1931-0145 (cited on page 75).

Brandes, Ulrik, Daniel Delling, Marco Gaertler, Robert Görke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner (2008). "On Modularity Clustering". In: *IEEE Transactions on Knowledge and Data Engineering* 20.2, pages 172–188. ISSN: 1041-4347 (cited on page 51).

Brin, Sergey and Lawrence Page (1998). "The anatomy of a large-scale hypertextual Web search engine". In: *Computer networks and ISDN systems* 30.1-7, pages 107–117 (cited on page 62).

Bringmann, Björn and Siegfried Nijssen (2008). "What Is Frequent in a Single Graph?" In: *Advances in Knowledge Discovery and Data Mining, 12th Pacific-Asia Conference, PAKDD 2008, Osaka, Japan, May 20-23, 2008 Proceedings.* Edited by Takashi Washio, Einoshin Suzuki, Kai Ming Ting, and Akihiro Inokuchi. Volume 5012. Lecture Notes in Computer Science. Springer, pages 858–863 (cited on page 60).

Calders, Toon, Bart Goethals, and Szymon Jaroszewicz (2006). "Mining rank-correlated sets of numerical attributes". In: *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, pages 96–105 (cited on pages 64, 65, 67, 70).

Cerf, Loïc, Jérémy Besson, Céline Robardet, and Jean-François Boulicaut (2008). "Data Peeler: Contraint-Based Closed Pattern Mining in n-ary Relations". In: *Proceedings of the SIAM International Conference on Data Mining, SDM 2008, April 24-26, 2008, Atlanta, Georgia, USA.* SIAM, pages 37–48 (cited on pages 8, 14, 21).

— (2009). "Closed Patterns Meet n-ary Relations". In: *ACM Transactions on Knowledge Discovery from Data* 3.1, pages 1–36 (cited on pages 8, 14, 21).

Clauset, Aaron, Mark Newman, and Cristopher Moore (2004). "Finding community structure in very large networks". In: *Physical Review E* 70.6, pages 1–6 (cited on page 51).

Crémilleux, Bruno and Arnaud Soulet (2008). "Discovering Knowledge from Local Patterns with Global Constraints". In: *Computational Science and Its Applications - ICCSA 2008, International Conference, Perugia, Italy, June 30 - July 3, 2008, Proceedings, Part II.* Edited by Osvaldo Gervasi, Beniamino Murgante, Antonio Laganà, David Taniar, Youngsong Mun, and Marina L. Gavrilova. Volume 5073. Lecture Notes in Computer Science. Springer, pages 1242–1257 (cited on pages 8, 33).

Cule, Boris, Bart Goethals, and Céline Robardet (2009). "A new constraint for mining sets in sequences". en. In: *Proc. SIAM Int. Conf. on Data Mining SDM'09*, pages 317–328 (cited on pages 8, 21, 26).

Desmier, Elise, Marc Plantevit, Céline Robardet, and Jean-François Boulicaut (2012). "Cohesive Co-evolution Patterns in Dynamic Attributed Graphs". In: *Discovery Science - 15th International Conference, DS 2012, Lyon, France, October 29-31, 2012. Proceedings.* Edited by Jean-Gabriel Ganascia, Philippe Lenca, and Jean-Marc Petit. Volume 7569. Lecture Notes in Computer Science. Springer, pages 110–124 (cited on pages 9, 83).

— (2013). "Trend Mining in Dynamic Attributed Graphs". In: *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2013, Prague, Czech Republic, September 23-27, 2013, Proceedings, Part I.* Edited by Hendrik Blockeel, Kristian Kersting, Siegfried Nijssen, and Filip Zelezný. Volume 8188. Lecture Notes in Computer Science. Springer, pages 654–669 (cited on pages 9, 59, 82).

Dong, Guozhu and Jinyan Li (1999). "Efficient Mining of Emerging Patterns: Discovering Trends and Differences". In: *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining.* KDD '99, pages 43–52 (cited on page 63).

Eagle, Nathan and Alex Pentland (2006). "Reality mining: Sensing complex social systems". In: *Journal of Personal and Ubiquitous Computing* 10.4, pages 255–268 (cited on page 36).

Echenique, Pablo, Jesus Gómez-Gardeñes, Yamir Moreno, and Alexei Vázquez (2005). "Distance-d covering problems in scale-free networks with degree correlations". In: *Physical Review E Rapid Communications* E 71 (cited on page 35).

Fayyad, Usama, Gregory Piatetsky-shapiro, and Padhraic Smyth (1996). "From Data Mining to Knowledge Discovery in Databases". In: *AI Magazine* 17, pages 37–54 (cited on page 7).

Fleury, Eric, Jean-Loup Guillaume, Céline Robardet, and Antoine Scherrer (2007). *Analysis of Dynamic Sensor Networks: Power Law Then What?* IEEE Conference on Communication System Software and Middleware (COMSWARE 2007) (cited on page 44).

Fortunato, Santo (2010). "Community detection in graphs". In: *Physics Reports* 486.3-5, pages 75–174 (cited on pages 50, 51, 55).

Freeman, Linton (1977). "A set of measures of centrality based on betweenness". In: *Sociometry* 40.1, pages 35–41 (cited on page 62).

Fürnkranz, Johannes and Arno J. Knobbe (2010). "Guest Editorial: Global modeling using local patterns". In: *Data Mining and Knowledge Discovery* 21.1, pages 1–8 (cited on page 33).

Ganter, Bernhard and Rudolf Wille (1997). *Formal Concept Analysis: Mathematical Foundations*. 1st. Secaucus, NJ, USA: Springer-Verlag New York, Inc. ISBN: 3540627715 (cited on page 8).

Garriga, Gemma C. (2003). "Discovering Unbounded Episodes in Sequential Data". In: *Knowledge Discovery in Databases: PKDD 2003, 7th European Conference on Principles and Practice of Knowledge Discovery in Databases, Cavtat-Dubrovnik, Croatia, September 22-26, 2003, Proceedings*. Edited by Nada Lavrac, Dragan Gamberger, Hendrik Blockeel, and Ljupco Todorovski. Volume 2838. Lecture Notes in Computer Science. Springer, pages 83–94 (cited on pages 26, 28, 29).

Georgii, Elisabeth, Koji Tsuda, and Bernhard Schölkopf (2011). "Multi-way set enumeration in weight tensors". In: *Machine Learning* 82.2, pages 123–155 (cited on page 13).

Giannotti, Fosca, Mirco Nanni, Fabio Pinelli, and Dino Pedreschi (2007). "Trajectory pattern mining". In: *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Jose, California, USA, August 12-15, 2007*. New York, NY, USA: ACM, pages 330–339. ISBN: 978-1-59593-609-7 (cited on page 75).

Hammond, David, Pierre Vandergheynst, and Rémi Gribonval (2011). "Wavelets on graphs via spectral graph theory". In: *Applied and Computational Harmonic Analysis* 30.2, pages 129–150 (cited on page 50).

Hamon, Ronan, Pierre Borgnat, Patrick Flandrin, and Céline Robardet (2013a). "Networks as Signals with an Application to a Bike Sharing System". In: *GlobalSIP 2013 Symposium on: Graph Signal Processing* (cited on pages 58, 93).

— (2013b). "Tracking of a dynamic graph using a signal theory approach: application to the study of a bike sharing system". In: *Proceedings of European Conference on Complex Systems (ECCS'13)* (cited on pages 9, 58, 93).

Hastings, W.K. (1970). "Monte Carlo Sampling Methods Using Markov Chains and Their Applications". In: *Biometrika* 57.1, pages 97–109 (cited on page 44).

Hero, Alfred and Bala Rajaratnam (2011). "Large Scale Correlation Screening". In: *Journal of the American Statistical Association* 106.496, pages 1540–1552 (cited on pages 55, 56).

Hirsch, Jorge (2005). "An index to quantify an individual's scientific research output". In: *Proceedings of the National Academy of Sciences of the United States of America* 102.46, page 16569 (cited on page 60).

Hui, Pan, Augustin Chaintreau, James Scott, Richard Gass, Jon Crowcroft, and Christophe Diot (2005). "Pocket Switched Networks and the Consequences of Human Mobility in Conference Environments". In: *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 244–251 (cited on pages 36, 37).

Ienco, Dino, Céline Robardet, Ruggero G. Pensa, and Rosa Meo (2013). "Parameter-less co-clustering for star-structured heterogeneous data". In: *Data Mining and Knowledge Discovery* 26.2, pages 217–254 (cited on page 7).

Jäschke, Robert, Andreas Hotho, Christoph Schmitz, Bernhard Ganter, and Gerd Stumme (2006). "TRIAS: An Algorithm for Mining Iceberg Tri-Lattices". In: IEEE Computer Society, pages 907–911 (cited on page 24).

Jensen, Pablo, Jean-Baptiste Rouquier, Nicolas Ovtracht, and Céline Robardet (2010). "Characterizing the speed and paths of shared bicycle use in Lyon". In: *Transportation Research Part D: Transport and Environment* 15.8, pages 522–524 (cited on pages 9, 48, 49).

Ji, Liping, Kian-Lee Tan, and Anthony K. H. Tung Anthony (2006). "Mining frequent closed cubes in 3D datasets". In: *Proceedings of the 32nd international conference on Very large data bases*. VLDB '06. Seoul, Korea: VLDB Endowment, pages 811–822 (cited on page 24).

Jiang, Daxin and Jian Pei (2009). "Mining frequent cross-graph quasi-cliques". In: *ACM Transactions on Knowledge Discovery from Data* 2.4, pages 1–42 (cited on page 60).

Kang, U, Charalampos Tsourakakis, Ana Paula Appel, Christos Faloutsos, and Jure Leskovec (2011). "HADI: Mining Radii of Large Graphs". In: *ACM Transactions on Knowledge Discovery from Data* 5.2, page 8 (cited on page 60).

Kendall, Maurice (1948). *Rank correlation methods*. London: Griffin. VII, 160 (cited on page 62).

Khan, Arijit, Xifeng Yan, and KunLung Wu (2010). "Towards proximity pattern mining in large graphs". In: *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2010, Indianapolis, Indiana, USA, June 6-10, 2010*. Edited by Ahmed K. Elmagarmid and Divyakant Agrawal. ACM, pages 867–878 (cited on page 60).

Lahiri, Mayank and Tanya Y. Berger-Wolf (2008). "Mining periodic behavior in dynamic social networks". In: *ICDM*. IEEE, pages 373–382 (cited on page 75).

Leicht, Elizabeth and Mark Newman (2008). "Community structure in directed networks". In: *Phys. Rev. Lett.* 100.11, page 118703 (cited on page 50).

Leyritz, Johan, Stéphane Schicklin, Sylvain Blachon, Céline Keime, Céline Robardet, Jean-François Boulicaut, Jérémy Besson, Ruggero Pensa, and Olivier Gandrillon (2008). "SQUAT: a web tool to mine human, murine, and avian SAGE data". In: *BMC Bioinformatics* 9.378 (cited on page 35).

Liu, Guimei and Limsoon Wong (2008). "Effective Pruning Techniques for Mining Quasi-Cliques". In: *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML/PKDD 2008, Antwerp, Belgium, September 15-19, 2008, Proceedings, Part II*. Edited by Walter Daelemans, Bart Goethals, and Katharina Morik. Volume 5212. Lecture Notes in Computer Science. Springer, pages 33–49 (cited on pages 60, 61).

Makino, Kazuhisa and Takeaki Uno (2004). "New Algorithms for Enumerating All Maximal Cliques". In: *Algorithm Theory - SWAT 2004, 9th Scandinavian Workshop on Algorithm Theory, Humlebaek, Denmark, July 8-10, 2004, Proceedings*, pages 260–272 (cited on page 60).

Mannila, Heikki and Hannu Toivonen (1997). "Levelwise Search and Borders of Theories in KnowledgeDiscovery". In: *Data Mining and Knowledge Discovery* 1.3, pages 241–258. ISSN: 1384-5810 (cited on page 11).

Mannila, Heikki, Hannu Toivonen, and A. Inkeri Verkamo (1997). "Discovery of Frequent Episodes in Event Sequences". In: *Data Mining and Knowledge Discovery* 1.3, pages 259–289. ISSN: 1384-5810 (cited on pages 26, 28).

Mannila, Heikki, Hannu Toivonen, and A. Inkeri Verkamo (1995). "Discovering frequent episodes in sequences (Extended Abstract)". In: *In 1st Conference on Knowledge Discovery and Data Mining*, pages 210–215 (cited on page 28).

Morishita, Shinichi and Jun Sese (2000). "Traversing Itemset Lattice with Statistical Metric Pruning". In: *Proceedings of the Nineteenth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, May 15-17, 2000, Dallas, Texas, USA*. Edited by Victor Vianu and Georg Gottlob. ACM, pages 226–236 (cited on page 14).

Mougel, Pierre-Nicolas, Christophe Rigotti, and Olivier Gandrillon (2012). "Finding collections of k-clique percolated components in attributed graphs". en. In: *PAKDD* (cited on page 60).

Negrevergne, Benjamin, Alexandre Termier, Marie-Christine Rousset, and Jean-François Mehaut (2013). "ParaMiner: a Generic Pattern Mining Algorithm for Multi-Core Architectures". In: *Journal of Data Mining and Knowledge Discovery (DMKD)*. Advance online publication. doi 10.1007/s10618-013-0313-2 (cited on page 33).

Newman, Mark (2004). "Fast algorithm for detecting community structure in networks". In: *Physical Review E* 69.066133 (cited on page 61).

Newman, Mark and Michelle Girvan (2004). "Finding and evaluating community structure in networks". In: *Physical Review* 69.0206113 (cited on page 50).

Pei, Jian and Jiawei Han (2000). "Can we push more constraints into frequent pattern mining?" In: *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, Boston, MA, USA, August 20-23, 2000*. Edited by Raghu Ramakrishnan, Salvatore J. Stolfo, Roberto J. Bayardo, and Ismail Parsa. ACM, pages 350–354 (cited on page 13).

Pensa, Ruggero G., Francesca Cordero, Céline Rouveirol, Rushed Kanawati, José A. Troyano, and Paolo Rosso, editors (2010). *Proceedings of the 1st Workshop on Dynamic Networks and Knowledge Discovery, Barcelona, Spain, September 24, 2010*. Volume 655. CEUR Workshop Proceedings. CEUR-WS.org (cited on page 82).

Pensa, Ruggero G., Céline Robardet, and Jean-François Boulicaut (2005). "A Bi-clustering Framework for Categorical Data". In: *Knowledge Discovery in Databases: PKDD 2005, 9th European Conference on Principles and Practice of Knowledge Discovery in Databases, Porto, Portugal, October 3-7, 2005, Proceedings*. Edited by Alípio Jorge, Luís Torgo, Pavel Brazdil, Rui Camacho, and João Gama. Volume 3721. Lecture Notes in Computer Science. Springer, pages 643–650 (cited on page 8).

Piatkowski, Nico, Sangkyun Lee, and Katharina Morik (2013). "Spatio-temporal random fields: compressible representation and distributed estimation". English. In: *Machine Learning* 93.1, pages 115–139. ISSN: 0885-6125 (cited on page 82).

Prado, Adriana, Marc Plantevit, Céline Robardet, and Jean-François Boulicaut (2013). "Mining Graph Topological Patterns: Finding Co-variations among Vertex Descriptors". en. In: *IEEE Transactions on Knowledge and Data Engineering*. In Press. Accepted for publication in July 2012. ISSN: ISSN: 1041-434 (cited on pages 9, 59, 60).

Preparata, Franco and Michael Shamos (1985). *Computational Geometry: An Introduction*. Springer-Verlag (cited on page 51).

Raedt, Luc De and Albrecht Zimmermann (2007). "Constraint-Based Pattern Set Mining". In: *Proceedings SIAM SDM'07*. Minneapolis, USA (cited on pages 8, 33, 75, 77).

Robardet, Céline (2002). "Contribution à la classification non supervisée: proposition d'une methode de bi-partitionnement". PhD thesis. Université Claude Bernard - Lyon 1 (cited on page 7).

— (2009). "Constraint-based Pattern Mining in Dynamic Graphs". In: *IEEE International Conference on Data Mining*. Edited by Philip S.Yu Sanjay Ranka, pages 950–955 (cited on pages 9, 59, 75).

Robardet, Céline, Vasile-Marian Scuturici, Marc Plantevit, and Antoine Fraboulet (2013). "When TEDDY meets GrizzLY: temporal dependency discovery for triggering road deicing operations". In: *The 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2013, Chicago, IL, USA, August 11-14, 2013*. Edited by Inderjit S. Dhillon, Yehuda Koren, Rayid Ghani, Ted E. Senator, Paul Bradley, Rajesh Parekh, Jingrui He, Robert L. Grossman, and Ramasamy Uthurusamy. ACM, pages 1490–1493 (cited on page 33).

Robert, Christian and George Casella (2004). *Monte Carlo Statistical Methods*. Springer (cited on page 44).

Salotti, Julien, Marc Plantevit, Céline Robardet, and Jean-François Boulicaut (2012). "Supporting the Discovery of Relevant Topological Patterns in Attributed Graphs". en. In: *Demo Session of the IEEE International Conference on Data Mining (IEEE ICDM 12)*. Edited by Jilles Vreeken, Charles Ling, Mohammed Zaki, Arno Siebes, Jeffrey Xu Yu, Bart Goethals, Geoffrey I. Webb, and Xindong Wu. IEEE Computer Society, pages 898–901 (cited on page 59).

Scherrer, Antoine, Pierre Borgnat, Eric Fleury, Jean-Loup Guillaume, and Céline Robardet (2008). "Description and simulation of mobility networks". In: *Computer Networks* 52.15, pages 2842–2858 (cited on pages 9, 36).

Silva, Arlei, Wagner Meira, and Mohammed Zaki (2010). "Structural Correlation Pattern Mining for Large Graphs". In: *Wks on Mining and Learning with Graphs* (cited on page 60).

— (2012). "Mining Attribute-structure Correlated Patterns in Large Attributed Graphs". In: *PVLDB* 5.5, pages 466–477 (cited on page 60).

Soulet, Arnaud and Bruno Crémilleux (2006). "Exploiting Virtual Patterns for Automatically Pruning the Search Space". In: *Knowledge Discovery in Inductive Databases, 4th International Workshop, KDID 2005, Porto, Portugal, October 3, 2005, Revised Selected and Invited Papers*. Edited by Francesco Bonchi and Jean-François Boulicaut. Volume 3933. Lecture Notes in Computer Science. Springer, pages 202–221 (cited on page 15).

Soulet, Arnaud, Chedy Raïssi, Marc Plantevit, and Bruno Crémilleux (2011). "Mining Dominant Patterns in the Sky". In: *11th IEEE International Conference on Data Mining, ICDM 2011, Vancouver, BC, Canada, December 11-14, 2011*. Edited by Diane J. Cook, Jian Pei, Wei Wang, Osmar R. Zaïane, and Xindong Wu. IEEE, pages 655–664 (cited on page 33).

Uno, Takeaki (2007). "An Efficient Algorithm for Enumerating Pseudo Cliques". In: *Algorithms and Computation, 18th International Symposium, ISAAC 2007, Sendai, Japan, December 17-19, 2007, Proceedings*. Volume 4835. Lecture Notes in Computer Science. Springer-Verlag, pages 402–414 (cited on page 77).

— (2010). "An Efficient Algorithm for Solving Pseudo Clique Enumeration Problem". In: *Algorithmica* 56.1, pages 3–16 (cited on pages 13, 18, 41, 60).

Watts, Duncan and Steven Strogatz (1998). "Collective dynamics of 'small-world' networks". In: *Nature* 393.6684, pages 440–442 (cited on page 35).

Wu, G (2000). "Frequency and Markov chain analysis of the amino-acid sequence of human alcohol dehydrogenase alpha-chain." In: *Alcohol Alcohol* 35.3, pages 302–306. ISSN: 0735-0414 (cited on page 32).

Zaki, Mohammed (2000). "Scalable Algorithms for Association Mining". In: *IEEE Trans. Knowl. Data Eng.* 12.3, pages 372–390 (cited on page 67).

Zhu, Feida, Xifeng Yan, Jiawei Han, and Philip S. Yu (2007). "GPrune: A Constraint Pushing Framework for Graph Pattern Mining". In: *Advances in Knowledge Discovery and Data Mining, 11th Pacific-Asia Conference, PAKDD 2007, Nanjing, China, May 22-25, 2007, Proceedings*. Edited by Zhi-Hua Zhou, Hang Li, and Qiang Yang. Volume 4426. Lecture Notes in Computer Science. Springer, pages 388–400. ISBN: 978-3-540-71700-3 (cited on page 76).

Constraint-based pattern mining approaches for the analysis of relational attributed dynamic graphs

Céline ROBARDET

In this Habilitation à Diriger des Recherches, I present the main results I have contributed to in the areas of local pattern extraction under constraints and of dynamic graph analysis.

Constraint-based pattern mining covers data mining algorithms that apply an exact search strategy to achieve the exhaustive extraction of the whole set of patterns satisfying some constraints over the data. These constraints are Boolean expressions based on evaluation criteria that measure the relevance of patterns in a specific data set. Besides, the use of constraints increases the computational efficiency of the process, making possible to truncate the search space while preserving the completeness of the extraction. After studying the constraint properties that have been identified as useful in this framework, I discuss the special case of formal concept extraction under constraints. Then I introduce the principles of a generic algorithm that 'pushes' constraints with various properties in the data mining system to optimize its efficiency.

Considering relational dynamic graphs, I present two case studies of mobility networks for which main global properties have been revealed thanks to statistical time-series analysis and clustering techniques.

In the last part, I discuss my main contributions on local pattern discovery in attributed and/or dynamic relational graphs. I first present an approach to characterize the relationship between vertex attributes and the graph topology in static attributed graphs. It consists in the extraction of co-variations between vertex attributes and measures describing the relationship of the vertex with the rest of the graph. Then, I propose to analyze dynamic graphs by discovering the main temporal changes as locally strong associations between vertices and their evolution through time. Finally, I introduce the mining of trends in attributed dynamic graphs to identify connected parts of the graph whose vertex attributes evolve in the same way.