Discovering Interesting Patterns in Large Graph Cubes

Florian Demesmaeker*†, Amine Ghrab†, Siegfried Nijssen* and Sabri Skhiri†
*Université Catholique de Louvain, Louvain-La-Neuve, Belgium
Email: siegfried.nijssen@uclouvain.be
†EURA NOVA R&D, Mont-Saint-Guibert, Belgium
Email: firstname.lastname@euranova.eu

***Abstract*—Due to the increasing importance and volume of highly interconnected data, such as in social or information networks, a plethora of graph mining techniques have been designed to enable the analysis of such data. In this work, we focus on the mining of associations between entity features in networks. We model each entity feature as a dimension to be analyzed. Consequently we build our approach on top of the existing graph cube framework which is an extension of the concept of the data cube to networks. Our task is particularly challenging because it requires the analysis of both the initial multidimensional graph and all its subsequent aggregate forms. As soon as we deal with a big data situation it is impossible for an analyst to consider manually all the possible views of the network data. The aim of this work is to design an algorithm for the discovery of interesting patterns in large graph cubes. Thus, instead of examining all the possible aggregations manually, the proposed technique leads the analyst to the interesting associations or *patterns* in the multidimensional graph.**

***Keywords*-Graph Mining, Graph Cubes, Frequent Itemset Mining**

I. Introduction

Due to the increasing availability of network data, various algorithms are being developed by the data mining community for the automated analysis of graphs. Moreover richer network information has become accessible, creating the need for graph mining techniques that consider both the network structure and the entities' features. In this work we search for interesting *patterns* in large graphs. For example let us say that we observe a high number of relationships between American and Chinese people. This is a pattern we may find in a social network. However, since a large number of people live in the USA and China, it is not surprising that the number of relationships between people of these two countries is important. On the other hand observing the same high number of connections between Belgian and Chinese people is more surprising. In this work we are looking for such surprising patterns over the characteristics of edges. To measure how surprising a pattern is, we define a *null* model that we assume has generated the data at hand.

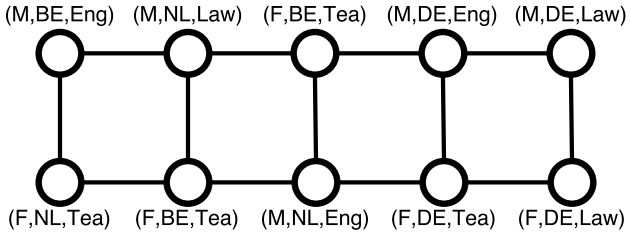
We model the graph of interest by a multidimensional network where nodes have attributes while edges have not. The graph cube framework proposed by Zhao et al. [21] is an extension of the data cube to graph data. From a multidimensional network this framework defines the different network aggregations, which are views of the original network without one or multiple attributes. In such an aggregate network nodes are merged in a way similar to a GROUP BY in a relational database and the weights of the edges are updated. Example 1 presents a multidimensional network and an aggregate network built from it.

Example 1. *Figure 1a presents our toy network that contains 10 nodes and 13 edges. Each node represents a person whose gender, location and profession are known. The location attribute can take three different values: Belgium (BE), The Netherlands (NL) or Germany (DE). Then the profession value is one of the following: Lawyer, Engineer or Teacher.*

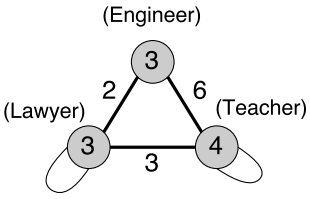
In this network multiple patterns along one or multiple attributes can be found. For instance we can consider [(Engineer), (Teacher)] or [(Lawyer, Lawyer)] that are two patterns defined over the Profession attribute only. But we can also be more specific by considering a pattern defined over the Location and Profession attributes, e.g. [(Belgium, Teacher), (Germany, Engineer)]. The two aggregations considered here are (Profession) and (Location, Profession) respectively. Figure 1b presents the aggregate network (Profession) where the location and the gender of the people are ignored. Section III introduces formally the graph cube and its aggregate networks.

A graph cube represents all the possible aggregations of the initial multidimensional graph. Graph cube mining is then no longer limited to the initial graph, but also explores all possible aggregations of the graph. Interesting patterns could be discovered inside each aggregate network. Each aggregation is built by considering or omitting every dimension of the original network. As a consequence the graph cube contains an exponential number of aggregate networks. In a big data setting, that is where the number of dimensions or the network itself is large, an analyst is overwhelmed by the number of patterns. Hence we propose an algorithm to help the analyst locate the interesting combinations of dimensions as well as the interesting patterns in these views.

The algorithms proposed in the state of the art allow the discovery of interesting patterns in a single graph mainly by analyzing its structure. In this work we consider both the structure of the graph and the internal information present in



(a) The toy network



(b) The aggregate network $(*, *, \text{Profession})$ built from the toy network

each node. We employ a statistical technique inspired from the frequent itemset mining literature to locate interesting patterns in large graph cubes. Thus, instead of examining the large cubes space manually, the proposed technique leads the analyst to the interesting pattern of the large multidimensional graph.

Existing algorithms that discover interesting itemsets include MINI [1] and MTV [2]. MINI searches for surprising itemsets while the MTV searches for representative itemsets. As MINI, our approach searches for surprising associations. Therefore we propose a mapping between pattern mining in multidimensional networks and frequent itemset mining, that we use to compare our approach with MINI. At the end of the day we consider two approaches: the original graph mining problem transformed so that MINI can be applied and the new graph approach that we propose.

The contribution of this paper could be summarized as follows:

- We propose a mapping between pattern mining in attributed graphs and frequent itemset mining in transactional databases.
- We formally define a measure of interestingness of patterns in a graph cube framework, and we provide an algorithm to search for the surprising patterns.
- We compare our method with an algorithm from the frequent itemset mining literature: MINI [1], and conduct experiments on two different datasets.

The remainder of this paper is organized as follows. Section II discusses the state of the art and presents the frequent itemset mining problem and existing algorithms to mine them. In Section III, we present the graph cube model and we propose the mapping from an attributed graph to a transactional database. Then we formally define a pattern within the graph cube data model in Section IV, and propose a corresponding measure of interestingness in Section V. Section VI presents the experiments we conducted on two different MovieLens datasets and how we built attributed graphs from these datasets. Finally we conclude this paper and discuss future work.

II. Related Work

At present existing algorithms discover surprising patterns in a graph mainly by analyzing its structure. For instance Akoglu et al. search for abnormal clusters of nodes in weighted graphs [3] by considering structural properties such as their densities. Other techniques, based on the minimum description length principle, include surprising substructures mining using Subdue [4] or anomaly mining by modeling a normative pattern [5]. Chakrabarti proposes Autopart [6] to detect outlier nodes by first reordering the adjacency matrix of a graph so that similar nodes are grouped together. However considering the features of the nodes brings more information and allows a refined analysis. Furthermore most approaches in the literature of attributed graph mining search for outlier nodes and not surprising features associations [7]. For instance Gao et al. present CODA [8], an algorithm to mine community outlier nodes by considering both node information and the network topology.

In the remainder of this section we briefly present the frequent itemset mining setting and the MINI algorithm that searches for surprising itemsets. As we shall see sets of attributes in a network can be seen as itemsets. Then we present another approach and the MTV algorithm that are based on the maximum entropy principle. This approach searches for itemsets that best describe the data and therefore is opposed to the hypothesis testing framework that we propose. To end this section we discuss an approach whose goal is highly similar to ours. It was recently proposed to find the most interesting aggregate networks in a graph cube.

Frequent itemset mining: The mining model, widely used in the frequent itemset literature, was first introduced by Agrawal et al. [9]. Given a set of transactions each containing a set of items, the task of a frequent itemset miner is to discover items that occur together in many transactions, and to build association rules. The number of transactions in which an itemset I is present is called the *support* of I. Since the number of subsets is exponential with its size, a prohibitively large number of frequent itemsets could be found. Therefore, an interesting phenomenon, called *pattern explosion*, arises with frequent itemset mining. Efficient algorithms were designed to mine frequent itemsets on large data. Agrawal and Srikant presented the Apriori algorithm [10] to perform such a task. Moreover, several condensed representations of the frequent itemsets have been proposed among which the maximal and the closed [11] itemsets. Algorithms using constraint programming have been proposed to mine efficiently closed [12] and maximal [13] frequent itemsets.

Surprising itemsets: However, an itemset that is frequent is not necessarily interesting. Gallo et al. presented MINI [1], an algorithm reporting the Most Informative and Non-redundant Itemsets. The idea behind MINI is to select itemsets that are hard to be explained by chance under prior knowledge on the data. Their algorithm takes as input the set of closed frequent itemsets of a database. Itemsets are qualified as informative if they are surprising with respect to some prior knowledge on the data. Moreover MINI ensures that overlapping itemsets are not both highly ranked as informative as they convey part of the same information.

Maximum entropy models and subjective interestingness: De Bie [17] presented a generic framework to represent prior information using the maximum entropy principle [14] by a probability distribution called MaxEnt. The entropy in the sense of Shannon [15] is a measure of uncertainty about a state. It can also be interpreted as the average quantity of information in this state or the number of bits needed to encode such information. Once the prior information has been used to build a probability distribution on the data, it is possible to evaluate the interestingness of a pattern.

The MaxEnt model can be used to quantify the interestingness of a pattern, for instance by considering its probability under MaxEnt. The negative log-probability is known as the self-information [18] in Shannon's information theory, the larger the more informative.

Succinctly summarizing data with itemsets: Mampaey et al. presented the MTV algorithm [2] to summarize data with itemsets. Their work is based on the generic framework of De Bie [17] to model the prior information using the maximum entropy principle. To assess the quality of a distribution model, Mampaey et al. proposed to use the Bayesian Information Criterion (BIC) [19] that favors models with fewer parameters. Given a set of itemsets, one can compute the maximum entropy model and its BIC score. The goal is to find the set that best summarizes the data, i.e. with the smallest score. This approach is not relevant to the goal of our graph miner because MTV is looking for representative itemsets while we are searching for surprising patterns in graphs. However, having a set of representative patterns induces that patterns not present in this set may be surprising.

Interesting features associations: Bleco and Kotidis [20] proposed an entropy-based filter to locate interesting node features associations in a graph. The aim is to locate automatically the relationships of interest among the exponential number of combinations. Their work iss based on the graph cube proposed by Zhao et al. [21]. Each aggregate network built from any combination of attributes is assigned an entropy value depicting the distribution of its nodes. However the intuition behind this choice is not clearly stated. Then the authors use an entropy-based measure to navigate within the cube lattice. This measure is used to define whether an aggregation level must be extended.

III. Preliminaries

A. Graph Cubes

First we introduce formally the concept of multidimensional network, a graph with attributes on the nodes. This is the data structure on which the graph cube is based. Then in Section IV we formally define the patterns we are looking for in such a network.

Definition 1. *A **multidimensional network** $\mathcal{N}$ is a graph denoted as $\mathcal{N} = (V, E, A)$, where V is a set of vertices, $E \subseteq V \times V$ is a set of edges and $A = \{A_1, A_2, \ldots, A_n\}$ is a set of n vertex-specific attributes, i.e. $\forall u \in V$, there is a multidimensional tuple $A(u) = (A_1(u), A_2(u), \ldots, A_n(u))$, where $A_i(u)$ is the value of the i-th attribute of the vertex u, with $1 \leq i \leq n$. A is called the dimension of the network $\mathcal{N}$ we consider.*

The toy network was presented as an example in Section I in Figure 1a. An aggregate network or *cuboid* built from it represents a view from such a network where one or multiple attributes are ignored. This view contains the information about a specific combination of attributes. We first introduce the notions of equivalence between nodes and edges used to define how to merge them to form cuboids. Then we give the formal definition of an aggregate network as introduced by Zhao et al. [21].

Definition 2. *Let $\mathcal{N} = (V, E, A)$ be a multidimensional network and $A' = (A'_1, A'_2, \ldots, A'_n)$ a possible aggregation of A, where $A'_i = A_i$ or $*$. If $A'_i = *$ then the i-th dimension is ignored in the process. Let $u, v \in V$ be two nodes of the network that are candidates to be equivalent. We say that u and v are **equivalent** according to A' if*

$$\forall i \text{ such that } A'_i \neq * : A'_i(u) = A'_i(v).$$

Furthermore we denote by $eq_{A'}(u, v)$ the function yielding a true value if u and v are equivalent according to A', or a false value otherwise.

As an example let us consider the aggregation $A' = (*, *, \text{Profession})$ and the toy network with numbered nodes and edges in Figure 3. Then the nodes 2, 5 and 10 are equivalent because they denote the three lawyers of the network.

Definition 3. *Let $\mathcal{N} = (V, E, A)$ be a multidimensional network and $A' = (A'_1, A'_2, \ldots, A'_n)$ a possible aggregation of A. Let $e, f \in E$ be two edges of the network such that $e = (u_e, v_e)$, $f = (u_f, v_f)$. We say that e and f are **equivalent** according to A' if their end nodes are equivalent, i.e.*

$$eq_{A'}(u_e, u_f) \wedge eq_{A'}(v_e, v_f) \vee eq_{A'}(u_e, v_f) \wedge eq_{A'}(u_f, v_e)$$

Similarly we denote by $eq_{A'}(e, f)$ the function yielding a true value if e and f are equivalent according to A', or a false value otherwise.

Considering $A' = (*, *, \text{Profession})$ in our toy example, e_2 and e_5 are two equivalent edges with respect to A' as they both associate an engineer and a lawyer.

Definition 4. *Let $\mathcal{N} = (V, E, A)$ be a multidimensional network and $A' = (A'_1, A'_2, \ldots, A'_n)$ a possible aggregation of A, where $A'_i = A_i$ or $*$. Then the **aggregate network** with respect to A' is a weighted graph $G' = (V', E')$, where*

1) *To every equivalence set of nodes V_{eq} of V, a node in the aggregate network $v' \in V'$ is associated. The weight of v' is the number of equivalent nodes in V_{eq}, i.e. $w(v') = |V_{eq}|$. Therefore v' is called a condensed vertex.*
2) *To every equivalence set of edges E_{eq} of E, an edge in the aggregate network $e' \in E'$ is associated. The weight of e' is the number of equivalent edges in E_{eq}, i.e. $w(e') = |E_{eq}|$. Therefore e' is called a condensed edge.*

From a multidimensional network multiple aggregations can be defined to obtain as many aggregate networks. One could wonder how these different networks are linked to each other. We introduce here the definition of a graph cube, based on [21].

Definition 5. *Given a multidimensional network $\mathcal{N} = (V, E, A)$, the **graph cube** is obtained by restructuring $\mathcal{N}$ in all possible aggregations of A. To each aggregation A' of A is associated an aggregate network G'. An aggregation of a multidimensional network $\mathcal{N} = (V, E, A)$ is called a **cuboid**.*

In the remainder of this paper we use the terms cuboid, graph and network to denote the multidimensional network defined by an aggregation. Figure 2 illustrates the concept of a graph cube lattice, that contains all the possible aggregate networks or cuboids. The numbers denote the number of nodes and edges in the networks. We say that a cuboid S' is an *ancestor* of another cuboid S if S is defined over all the attributes of S' and one or multiple other attributes. For example $(*, *, \text{Profession})$ is an ancestor of $(*, \text{Location}, \text{Profession})$. The most aggregated network represented at the top of the lattice is called the *apex*. It is the ancestor of all the cuboids. Furthermore we call S a *direct ancestor* if it is defined over all the attributes

B. Mapping Pattern Mining in Graphs to Itemset Mining

In this work, we propose one approach that maps the problem of finding interesting aggregate edges in the graph cube to the problem of finding interesting itemsets in a transactional database. The idea is that each edge in the original graph can be mapped to a transaction and a pair of attribute values can be mapped to an item. Hence a transaction consisting of a set of items is equivalent to a set of pairs of attribute values defining an edge.

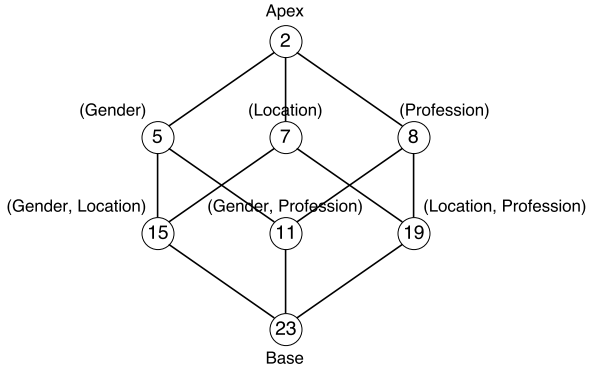


Figure 2: Graph cube lattice

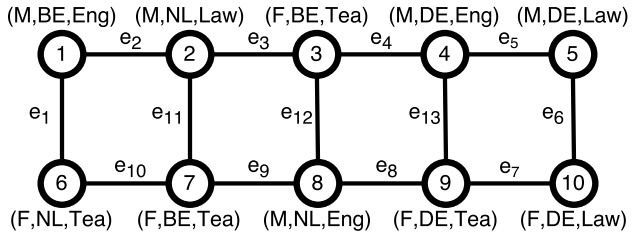


Figure 3: The toy network with numbered nodes and edges

Let $\mathcal{N} = (V, E, A)$ be a multidimensional network. Any edge $e \in E$ can be translated into a transaction $X_1, \ldots, X_n$ with $n = |A|$ is the number of attributes in $\mathcal{N}$. Indeed an undirected edge e is fully described by the attributes of the two nodes associated to e. Definition 6 formalizes the mapping.

Definition 6. *Let $\mathcal{N} = (V, E, A)$ be a multidimensional network and $u, v \in V, e \in E : e = (u, v)$. Let $\mathcal{A}$ be the set of all possible node attribute tuples. Let $\mathcal{T}$ be the set of all possible transactions. The **mapping** $M : \mathcal{A}^2 \mapsto \mathcal{T}^2$ takes a pair of tuples representing an edge as input and returns the corresponding pair of transactions as output. Thus the two transactions corresponding to e are given by*

$$M(A(u), A(v)) = (\{X_1, \ldots, X_{|A|}\}, \{Y_1, \ldots, Y_{|A|}\})$$

with $X_i = (A_i(u), A_i(v)), Y_i = (A_i(v), A_i(u)) \quad \forall i$

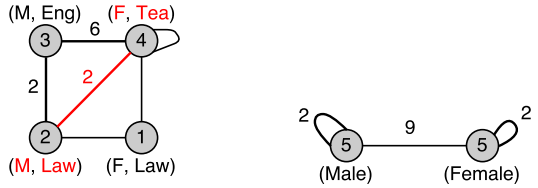
Since we deal with undirected graphs, the mapping we defined above yields a transactional database that is more consistent with the original attributed graph because it maps two transactions to a single edge. Figure 3 and Table I present respectively our toy network and the mapping between the edges and their corresponding pairs of transactions.

IV. Frequent Pattern Mining

This section defines the kind of patterns we are looking for in the data as well as some measures related to these patterns. We consider that a pattern is an edge of any graph in the graph cube lattice as stated in Definition 7 and illustrated

Edge	Tuples	Transactions	Edge	Tuples	Transactions
e_1	(F, NL, T), (M, BE, E)	{FM, NLBE, TE}, {MF, BENL, ET}	e_8	(F, DE, T), (M, NL, E)	{FM, DENL, TE}, {MF, NLDE, ET}
e_2	(M, BE, E), (M, NL, L)	{MM, BENL, EL}, {MM, NLBE, LE}	e_9	(M, NL, E), (F, BE, T)	{MF, NLBE, ET}, {FM, BENL, TE}
e_3	(M, NL, L), (F, BE, T)	{MF, NLBE, LT}, {FM, BENL, TL}	e_{10}	(F, BE, T), (F, NL, T)	{FF, BENL, TT}, {FF, NLBE, TT}
e_4	(F, BE, T), (M, DE, E)	{FM, BEDE, TE}, {MF, DEBE, ET}	e_{11}	(M, NL, L), (F, BE, T)	{MF, NLBE, LT}, {FM, BENL, TL}
e_5	(M, DE, E), (M, DE, L)	{MM, DEDE, EL}, {MM, DEDE, LE}	e_{12}	(F, BE, T), (M, NL, E)	{FM, BENL, TE}, {MF, NLBE, ET}
e_6	(M, DE, L), (F, DE, L)	{MF, DEDE, LL}, {FM, DEDE, LL}	e_{13}	(M, DE, E), (F, DE, T)	{MF, DEDE, ET}, {FM, DEDE, TE}
e_7	(F, DE, L), (F, DE, T)	{FF, DEDE, LT}, {FF, DEDE, TL}			

Table I: Mapping from the toy network to its corresponding transactional database



(a) Aggregate network (Gender, $*$, Profession)

(b) Aggregate network (Gender, $*$, $*$)

Figure 4: The (Gender, $*$, $*$) and (Gender, $*$, Profession) aggregate networks built from the toy network

in Example 2. In Section V we define the interestingness of a pattern P to be related to the probability to observe it as many times as it appears in the data. This probability is computed given the prior knowledge contained in the ancestors of the graph where the pattern lies.

The prior knowledge we will use consists of the direct ancestors of the cuboid where the pattern lies. Consequently we define S to be an arbitrary selection of attributes such that $S \subseteq A$ contains at least an element. In fact we can define S as being any cuboid in the graph cube lattice except the apex, the most aggregated graph containing a single node and a potential self-loop.

Definition 7. *In a network $\mathcal{N} = (V, E, A)$, a **pattern** P is defined by a selection of attributes $S \subseteq A$ and two tuples of values T_1 and T_2 defined over the attributes in S. The pattern P is then an edge (u, v) such that $S(u) = T_1, S(v) = T_2$ with $u, v \in V$.*

Example 2. *Considering our toy network, we can define as many patterns as the total number of edges in all the graphs of the graph cube lattice but the apex. Figure 4a highlights the pattern [T_1, T_2] = [(Male, Lawyer), (Female, Teacher)] denoted by P which is present two times in the data. P is defined upon the cuboid or subset of attributes $S_1 = (Gender, *, Profession)$. It corresponds to the two edges e_3 and e_{11} in the original graph in Figure 3.*

Similarly to frequent itemset mining, we denote the number of occurrences of P as its *support* and formalize it as follows.

Definition 8. *Let $\mathcal{N} = (V, E, A)$ be a multidimensional network, $P = (T_1, T_2)$ be a pattern defined over an aggregation $S \subseteq A$, and $e_P = (u, v)$ be an edge such that $S(u) = T_1$ and $S(v) = T_2$. Then the **support** $supp_S(T_1, T_2)$ of P is equal to*

$$\sum_{e \in E \ : \ eq_S(e, e_P)} 1$$

Please note that the support of a pattern in a cuboid is the weight of the pattern in such a network. For example, in Figure 4a the support of the pattern [(Female, Lawyer), (Female, Teacher)] is 1 and in Figure 4b the support of [(Male), (Female)] is 9. Both figures denote aggregate networks built from the toy example. In Section V we define the interestingness of a pattern based on its support.

V. Surprising Patterns Discovery

To quantify how surprising a pattern is we use an hypothesis testing framework. For a particular pattern P we compute the probability to observe $supp(P)$ occurrences of it, given some prior knowledge we have on the data. This knowledge is based on the ancestors of the network where P lies. In this section we model the prior information and then compute the conditional probability of observing $supp(P)$ occurrences of P.

A. Pattern probability

First we define the probability of a node having certain attributes defined by a tuple T with respect to a more aggregated view of the network and denote it by p. The probability $p_{S,S'}(T)$ of a node having attribute values T over S is the fraction of nodes having attribute values T among the nodes having attribute values T' where T' is defined over $S' \subset S$. Therefore p is the probability $\Pr(T \mid T')$.

Definition 9. *The **proportion** $p_{S,S'}(T)$ of a tuple of values T defined over a set of attributes S with respect to a set of attributes $S' \subset S$ is given by*

$$\frac{supp_{S'}(T_1, T_2)}{supp_S(T_1, T_2)}$$

Example 3. *Let us consider again our toy network. Figure 4a and Figure 4b present the networks associated with the cuboids $S = (Gender, *, Profession)$ and $S' = (Gender, *, *)$ respectively. Then the proportion of the tuple (Male, Lawyer) with respect to S' is given by*

$$p_{S,S'}[(M,\ Law)] = \frac{W_V[(M, Law)]}{W_{V'}[(M)]} = \frac{2}{5}$$

where W_V and $W_{V'}$ are the functions mapping a tuple of attributes to its weight in the networks corresponding to the cuboids S and S' respectively.

A possible definition for the probability of a pattern is inspired by the partition models [22] and adapted to the graph cube lattice we are working with. The idea in the itemset mining setting is to assume that all the elements of a partition of the items occur independently from each other. Here we consider an independence model, i.e. a particular case of a partition model.

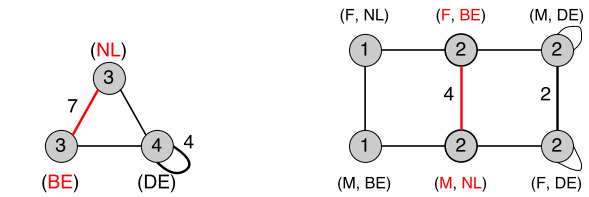
In our setting let $P' = (T'_1, T'_2)$ be the pattern P restricted to S', i.e. P from which we remove the values of the attributes $S \setminus S'$. The idea is to compute the probability of observing P given that we observe P' a certain number of times. We calculate this probability under the null hypothesis that the data is generated from an *independence* model. In this model we assume that each node has the same probability to be seen. As a consequence the probability of observing an edge $P = (T_1, T_2)$ is the product of the proportions of its two end points T_1 and T_2 with respect to S'. If we take an edge at random in G, the probability of the end points attribute values to match the pattern P given that we know the support of P' is given by Equation 1.

$$\Pr(P \mid G') = p_{S,S'}(T_1) p_{S,S'}(T_2) \tag{1}$$

This is the probability that one of the $supp(P')$ edges is an edge between T_1 and T_2. P has a single equivalent edge in G'. This is the edge whose nodes have the same properties as the nodes of P except the aggregated one(s). Several edges in G can have the same ancestor edge in G'. Therefore we can define the probability to draw an edge P from its corresponding ancestor edge which has a support greater or equal to the support of P.

Example 4. *Let us onsider $P' = [(Male), (Female)]$ highlighted in Figure 4b and Figure 5b. It is distributed into 5 aggregate edges in the cuboid (Gender, Location, $*$). Each edge has its own probability to be drawn from the 9 edges of the cuboid (Gender, $*$, $*$). For instance the probability that the edge $P_1 = [(Female, BE), (Male, NL))$ is drawn from the 9 edges is given by $\frac{2}{5} \times \frac{2}{5} = 0.16$. Please note that $supp(P_1) = 4$ and that P_1 is as likely to be drawn as $P_2 = ((Female, BE), (Male, DE))$ while $supp(P_2) = 1$.*

However this quantity is not always correct. Let us consider the $(\text{Gender}, *, *)$ network in Figure 4b that contains a total of 10 nodes and 13 edges. Its only ancestor is the apex that contains a self-loop. Considering that we draw an edge from one of the 13 edges aggregated in the self-loop, the probability to get the edge [(Male), (Female)] is equal to the product of the proportions of (Male) and (Female), that is 0.25. The two other edges of this network have the same probability to be drawn, leading to a sum of probabilities equal to 0.75. The missing fourth comes from the [(Male), (Female)] edge. Indeed as we draw an edge by drawing its two end points successively we should add the probability to draw a (Female) node after a (Male) node. Our drawing process induces an order between the nodes and therefore a direction to the edges.



(a) Aggregate network $(*, \text{Location}, *)$

(b) Aggregate network $(\text{Gender}, \text{Location}, *)$

Figure 5: The networks corresponding to the cuboids $(*, \text{Location}, *)$ and $(\text{Gender}, \text{Location}, *)$ built from the toy network

Generally speaking this situation occurs when drawing a non self-loop edge (T_1, T_2) whose equivalent edge in an ancestor cuboid is a self-loop. Hence we choose to define the probability of observing an undirected edge (T_1, T_2) as the sum of the probability to draw T_1 then T_2 and the probability to draw T_2 then T_1. This means that in the toy example we mentioned above we should add the probability to draw a man then a woman and the probability to draw a woman then a man when drawing from the unique self-loop edge of the apex. Hence the probability to draw an edge between a man and a woman is equal to 0.5. This solution induces a bias towards non self-loops being more likely than self-loops.

B. Pattern interest

The interest of a pattern can be defined in various ways. In this work we say that a pattern is interesting if it is surprising with respect to some prior knowledge. We derive the probability that a pattern P has indeed a support equal to the observed support $supp(P)$, given that P is drawn from the aggregate edge in the ancestor G' equivalent to P. We model the support of P as a random variable following a binomial distribution. The Bernoulli trial is the following: an edge is drawn from the aggregate edge in G' equivalent to P with probability $p = \Pr(P \mid G')$ as given in Equation 1.

Equation 2 defines the probability to observe $supp(P)$ occurrences of P in a cuboid G defined over S with respect to an ancestor cuboid $S' \subset S$. The multidimensional network corresponding to S' is given by $G' = (V', E', A')$.

$$\Pr(supp(P)) = \binom{|E'|}{supp(P)} p^{supp(P)} (1-p)^{|E'| - supp(P)} \tag{2}$$

The components of this equation are the following.

- The binomial coefficient is the number of possibilities to draw the actual number of edges from the number of aggregate edges.

- The number of edges $|E'|$ is equal to $supp(P')$.
- p is the probability to draw P from P'.
- $1-p$ is the probability to draw another edge from P'.

Example 5. *Let us consider the pattern* $P = [(\textit{Female, BE}), (\textit{Male, NL})]$ *whose context is given in Figure 5. The probability of* P *given the (Gender,* $*$*,* $*$*) network given in Figure 4b was computed in Example 4 and is given by* $p = 0.16$*. The probability to effectively observe* 4 *such edges is equal to*

$$\binom{9}{4} 0.16^4 0.84^5 = 0.003.$$

On the other hand the probability to observe the 4 *edges of* P *according to the aggregation (*$*$*, Location,* $*$*) whose network is given in Figure 5a is equal to*

$$\binom{7}{4} 0.36^4 0.64^3 = 0.154.$$

Given each ancestor $S' \subseteq S$ we can compute a probability by Equation 2. To avoid considering the exponential number of ancestors $S' \subseteq S$ we restrict ourselves to the direct ancestors of S denoted by S_{anc}. For every $S' \in S_{anc}$, we compute the probability $p = \Pr(supp(P) \mid S')$. The ancestors S_{anc} and the independence model form the prior knowledge on the data. If one of the ancestors yields a high probability p, then this part of the prior knowledge implies the actual support of P under the independence model. On the other hand if all the ancestors yield a small probability p, then nothing in the prior knowledge can explain why we observe $supp(P)$. Hence we define the interestingness of a pattern to be equal to the highest probability computed from its ancestors. We define the interestingness of a pattern as the quantity that best *explains* $supp(P)$ according to the independence model and denote it by $int(P)$.

$$int(P) = \max_{S' \in S_{anc}} \Pr(supp(P)) \tag{3}$$

We calculated the probability of observing $supp(P)$ for $P = [(\text{Female, BE}), (\text{Male, NL})]$ given each of its direct ancestor networks in Example 5. This pattern is best explained by the ($*$, Location, $*$) network with a probability of 0.15. Hence the interestingness of P is equal to 0.15.

C. Search algorithm

Algorithm 1 performs an exhaustive search throughout every aggregate network of the multidimensional network $\mathcal{N}$ using a probability threshold θ. Any aggregate edge having the probability of its support above this threshold is not interesting.

Algorithm 1: Search for surprising patterns in the graph cube lattice

Input : $\mathcal{L}$ the lattice containing all the aggregate networks built from a given network
Input : θ the aggregate edge probability threshold
Output: The set of surprising aggregate edges in any network of $\mathcal{L}$ with respect to θ

```
patterns ← ∅
/* Most aggregated graph is apex */
for 𝒩 ← ℒ \ apex do
    (V, E, A) ← 𝒩
    for e ∈ E do
        interesting = true
        ancestors =
        {(V', E', A') | A' ∈ A, |A'| = |A| − 1}
        for 𝒩' ← ancestors do
            p = probability(𝒩, 𝒩', e)
            if p ≥ θ then
                interesting = false
                break
            end
        end
        if interesting then
            patterns ← patterns ∪ (e, p)
        end
    end
end
return patterns sorted by probability
```

VI. Experiments

We test our measure of interestingness in attributed graphs on two different datasets. The first one is collected by the GroupLens research lab at the University of Minnesota[1] and consists of a million of ratings given by MovieLens[2] users to movies. Demographic information on the users is also provided. We use the dataset to build a graph where the nodes correspond to users and we create an edge between two users if they have the same cinematographic tastes.

The second dataset is an enhancement of the 10 millions ratings MovieLens dataset where information about the movies from the Internet Movie Database[3] (IMDb) and Rotten Tomatoes[4] RT has been added. Therefore we build a graph where a node represents a movie and there is an edge between two movies if they are liked by a certain number of same users. We present in this section how we construct these two attributed graphs and the experiments we conducted on them.

A. Graph construction

The MovieLens dataset: The MovieLens1M dataset encompasses some demographic information about the users, namely their age, gender, occupation and location. The

[1]GroupLens Research group, https://www.grouplens.org

[2]MovieLens, movie recommendations, https://www.movielens.org

[3]Internet Movie Database, http://www.imdb.com

[4]Rotten Tomatoes - movie critic reviews, https://www.rottentomatoes.com

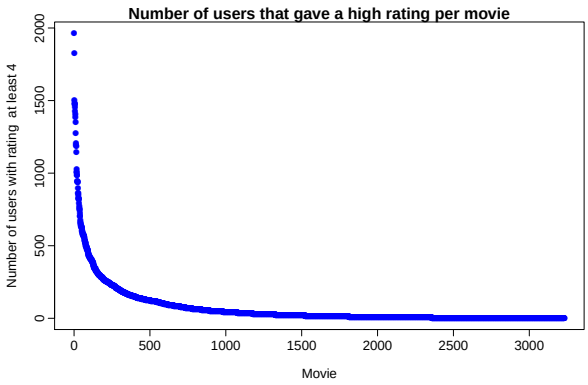


Figure 6: Number of high ratings per movie in the MovieLens1M dataset

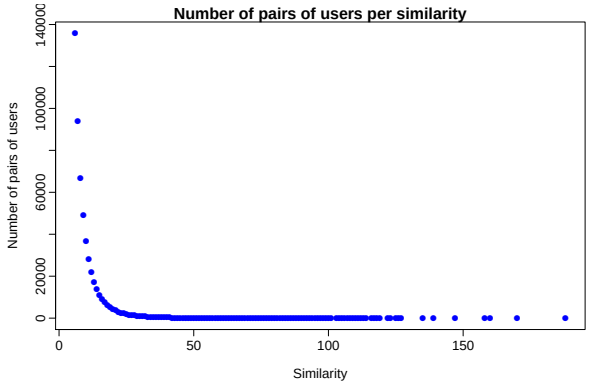


Figure 7: Number of pairs of users per similarity in the pruned MovieLens1M dataset

Dataset	**Users**	**Movies**	**User features**	**Ratings**
MovieLens1M	6000	3700	(age, gender, occ., loc.)	1,000,000
Network	**Users**	**Nodes**	**Similarities**	**Edges**
$\mathcal{N}_1$	1393	975	32537	27150

Table II: Statistics about the MovieLens1M dataset and network $\mathcal{N}_1$ built from it

dataset consists of a million of ratings given between 2000 and 2003, its characteristics are given in Table II. The rating scale is an integer from 1 to 5 stars. We include users that gave at least 20 ratings. From this dataset we build a multidimensional network $\mathcal{N}_1 = (V_1, E_1, A_1)$. The set of nodes V_1 is directly the set of provided users. The attributes of the nodes A_1 are the user features provided in the dataset. The `age` of a user belongs to one of the 7 age groups while there are 21 possible occupations. The location is given as a zip-code and we categorized it into U.S. states. The set of edges E_1 requires a similarity measure between users to be built.

We choose to put an edge between two users if they like a certain number of movies in common. We arbitrarily say that a user likes a movie if she rates it 4 or above . We define the similarity between two users to be the number of movies they both rated 4 stars or more. However one can expect from a movie dataset that the function depicting the number of high ratings per movie follows a power law, a few blockbusters having a lot of high ratings while the rest of the movies is highly less rated at all. Figure 6 presents such a function. As expected the curve follows a power law and we can observe the long tail of the distribution.

Since the blockbusters will favor a higher similarity between all the users we choose to remove them. The function presented in Figure 6 shows an abrupt slope near the 500 users. Therefore we consider as blockbuster in this dataset the movies having at least 500 ratings of 4 stars or more and remove them. We refer to the obtained dataset as the pruned MovieLens1M dataset. Then we need to define a threshold from which the similarity measure yields an edge in the network. For this purpose we show in Figure 7 the relation between a similarity and the number of pairs of users having this similarity. We removed similarities 1 to 5 as they have a very high amount of pairs of users in order to observe the curve in more details.

We observe that this relation also follows a power law, a high number of pairs having a low similarity while a few pairs have a high similarity. We also observe that the slope becomes abrupt around a similarity of 20. Therefore in order to reduce the number of edges in the newly created network we set the similarity threshold to 20. The statistics about the obtained network $\mathcal{N}_1$ are given in Table II.

The extended MovieLens dataset: The second International Workshop on Information Heterogeneity and Fusion in Recommender Systems [23] (HetRec 2011) released several datasets among which the extended MovieLens dataset. It consists of the MovieLens10M dataset enhanced by more information about the movies from IMDb and RT such as the country of origin or the different user and critics ratings. Statistics about this dataset are presented in Table III The ratings were given from 1995 to 2009 on the MovieLens website and range from 0.5 to 5 stars.

For this network $\mathcal{N}_2 = (V_2, E_2, A_2)$ we model the nodes V_2 to be the movies. Each movie has three attributes: its year of release, its country of origin and the average top critic. The original average RT top critic is given on a scale from 0 to 10 but we categorize it into 5 classes: very bad, bad, average, good, very good. We also categorize the year of release by replacing them by their respective decade. Finally we have $A_2 = (\texttt{critic}, \texttt{country}, \texttt{decade})$. As for the MovieLens1M dataset, we need a similarity measure to build the set of edges E_2.

Similarly to the approach described above, an edge is put between two movies if they have a certain number of users that gave them a rating of 4 stars or more in common. The function depicting the number of movies rated 4 or more per user also follows a power law. A few users took the time to rate a high number of movies or like a lot of them while

Dataset	Users	Movies	Movie features	Ratings
HetRec11	2100	10,200	(critic, country, decade)	860,000
Network	**Movies**	**Nodes**	**Similarities**	**Edges**
$\mathcal{N}_2$	678	318	25808	10985

Table III: Statistics about the extended MovieLens10M dataset and network $\mathcal{N}_2$ built from it

Cuboid	Pattern	Probability
(Gender)	[(M), (M)]	0
(Gender)	[(M), (F)]	2.31×10^{-292}
(Age)	[(35), (18)]	1.4×10^{-71}
(State)	[(GU), (CA)]	3.88×10^{-67}
(Gender)	[(F), (F)]	2.1×10^{-65}
(Age, State)	[(50, PA), (25, CA)]	1.95×10^{-47}
(Age)	[(18), (50)]	3.12×10^{-47}
(State)	[(CA), (LA)]	1.75×10^{-44}
(Age)	[(18), (56)]	1.09×10^{-40}
(Age)	[(25), (25)]	9.42×10^{-40}

Table IV: Top 10 most interesting patterns in the MovieLens1M dataset reported by our miner

most users express their preferences for a small number of movies

Therefore a few users are easily similar to a lot of other users by liking a high number of movies. We decide to remove them by not taking into account users that gave a high rating to more than 400 movies. Then the similarity threshold is empirically defined in a similar manner as we did for the MovieLens1M dataset. We set the similarity threshold to 40. The statistics about the obtained network $\mathcal{N}_2$ are given in Table III.

B. Results

In this section we present the patterns output by our miner on the two MovieLens datasets. We interpret the results of our miner on both datasets and we show the limitations of the MINI algorithm.

Regarding the MovieLens1M dataset where nodes are users the 10 most interesting edges according to the probability measure are given in Table IV. We can observe that the three aggregate edges of the `gender` cuboid are output. It seems to be the case that attributes having a small set of values are more likely to be surprising.

As for the extended MovieLens10M dataset where nodes are movies the 10 most interesting edges are presented in Table V. We can observe that 4 of the edges are between top critic values. It seems to confirm the fact that attributes with a small set of values are more likely to be output as surprising. A possible interpretation of these results is the following. As the second most interesting edge lies between *very bad* and *good* movies according to the average top critic ratings, we can infer that people rating the movies in this dataset do not agree with the top critics.

We also ran the MINI algorithm on the two datasets. Results are given in Table VI and Table VII. As the algorithm uses a greedy heuristic to add the itemsets to the set of interesting patterns, we set the maximum number of iterations to $10,000$. On both datasets the results are not really interesting, MINI reports the relations that often occur and that have a relatively high p-value. Moreover MINI did not return 10 patterns. Better results can be obtained by increasing the maximum number of iterations. But we report here only these results to illustrate the fact that MINI is greedy and hence can be fooled by unsurprising but frequent patterns.

Cuboid	Pattern	Probability
(Critic)	[(Good), (Very good)]	3.67×10^{-194}
(Critic)	[(Good), (Very bad)]	4.19×10^{-134}
(Critic)	[(Very good), (Very good)]	4.19×10^{-130}
(Country)	[(USA), (USA)]	5.32×10^{-117}
(Critic)	[(Average), (Very bad)]	9.52×10^{-106}
(Critic)	[(Very bad), (Very bad)]	1.04×10^{-82}
(Country, Critic)	[(USA, Good), (USA, Good)]	3.13×10^{-72}
(Critic)	[(Good), (Good)]	1.16×10^{-64}
(Country)	[(USA), (Spain)]	8.91×10^{-61}
(Country)	[(Ireland), (USA)]	7.62×10^{-59}

Table V: Top 10 most interesting patterns in the extended MovieLens10M dataset reported by our miner

Cuboid	Pattern	p-value
(Age)	[(25), (25)]	0.74
(Gender)	[(M), (F)]	0.74
(Gender)	[(F), (M)]	0.74
(Gender)	[(M), (M)]	0.74

Table VI: Top 10 most interesting patterns in the Movielens1M dataset reported by MINI

Cuboid	Pattern	p-value
(Country, Critic)	[(USA, Good), (USA, Good)]	8.85×10^{-11}
(Critic)	[(Good), (Very good)]	0.74
(Critic)	[(Very good), (Good)]	0.74
(Decade)	[(2000), (2000)]	0.74
(Decade)	[(2000), (1990)]	0.74
(Decade)	[(1990), (1990)]	0.74

Table VII: Top 10 most interesting patterns in the extended Movielens10M dataset reported by MINI

VII. Conclusion

Our objective was to study the patterns that can be found in attributed graphs. We based our work on the graph cube data model and proposed a hypothesis testing framework to evaluate how surprising the found patterns are with respect to an independence model. We showed the relationship between our pattern mining framework and the frequent itemset mining literature. Moreover we proposed a mapping from attributed graphs to transactional databases. We compared the frequent itemset mining algorithm MINI with our method theoretically. Furthermore we gave the

interpretability of the results obtained on two MovieLens datasets with MINI and with our method.

Many opportunities for extending this work exist. Experiments on synthetic datasets and other real datasets can be conducted to understand in more depth the patterns that can be found in the graph cube lattice. The maximum entropy model on binary databases could be adapted to attributed graph data to provide a generic framework for mining patterns in networks. The notion of pattern can be extended to heterogeneous features associations. For instance a pattern could consist of the number of relationships between Belgians and engineers.

REFERENCES

[1] A. Gallo, T. De Bie, and N. Cristianini, "Mini: Mining informative non-redundant itemsets," in *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, 2007, pp. 438–445.

[2] M. Mampaey, N. Tatti, and J. Vreeken, "Tell me what i need to know: succinctly summarizing data with itemsets," in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2011, pp. 573–581.

[3] L. Akoglu, M. McGlohon, and C. Faloutsos, "Oddball: Spotting anomalies in weighted graphs," *Advances in Knowledge Discovery and Data Mining*, pp. 410–421, 2010.

[4] C. C. Noble and D. J. Cook, "Graph-based anomaly detection," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2003, pp. 631–636.

[5] W. Eberle and L. Holder, "Anomaly detection in data represented as graphs," *Intelligent Data Analysis*, vol. 11, no. 6, pp. 663–689, 2007.

[6] D. Chakrabarti, "Autopart: Parameter-free graph partitioning and outlier detection," in *European Conference on Principles of Data Mining and Knowledge Discovery*. Springer, 2004, pp. 112–124.

[7] L. Akoglu, H. Tong, and D. Koutra, "Graph based anomaly detection and description: a survey," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 626–688, 2015.

[8] J. Gao, F. Liang, W. Fan, C. Wang, Y. Sun, and J. Han, "On community outliers and their efficient detection in information networks," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 813–822.

[9] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *Acm sigmod record*, vol. 22, no. 2. ACM, 1993, pp. 207–216.

[10] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen, A. I. Verkamo *et al.*, "Fast discovery of association rules." *Advances in knowledge discovery and data mining*, vol. 12, no. 1, pp. 307–328, 1996.

[11] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal, "Discovering frequent closed itemsets for association rules," in *International Conference on Database Theory*. Springer, 1999, pp. 398–416.

[12] T. Uno, T. Asai, Y. Uchida, and H. Arimura, "Lcm: An efficient algorithm for enumerating frequent closed item sets." in *FIMI*, vol. 90. Citeseer, 2003.

[13] D. Burdick, M. Calimlim, and J. Gehrke, "Mafia: A maximal frequent itemset algorithm for transactional databases," in *Data Engineering, 2001. Proceedings. 17th International Conference on*. IEEE, 2001, pp. 443–452.

[14] E. T. Jaynes, "Information theory and statistical mechanics," *Physical review*, vol. 106, no. 4, p. 620, 1957.

[15] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 5, no. 1, pp. 3–55, 2001.

[16] J. Han, J. Pei, and M. Kamber, *Data mining: concepts and techniques*. Elsevier, 2011.

[17] T. De Bie, "Maximum entropy models and subjective interestingness: an application to tiles in binary databases," *Data Mining and Knowledge Discovery*, vol. 23, no. 3, pp. 407–446, 2011.

[18] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 1991.

[19] G. Schwarz *et al.*, "Estimating the dimension of a model," *The annals of statistics*, vol. 6, no. 2, pp. 461–464, 1978.

[20] D. Bleco and Y. Kotidis, "Entropy-based selection of graph cuboids," in *Proceedings of the Fifth International Workshop on Graph Data-management Experiences & Systems*. ACM, 2017, pp. 2:1–2:6.

[21] P. Zhao, X. Li, D. Xin, and J. Han, "Graph cube: on warehousing and olap multidimensional networks," in *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. ACM, 2011, pp. 853–864.

[22] J. Vreeken and N. Tatti, "Interesting patterns," in *Frequent pattern mining*. Springer, 2014, pp. 105–134.

[23] I. Cantador, P. Brusilovsky, and T. Kuflik, "2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011)," in *Proceedings of the 5th ACM conference on Recommender systems*, ser. RecSys 2011. New York, NY, USA: ACM, 2011.