UCL Université catholique de Louvain

École polytechnique de Louvain (EPL)

epl

Graph Cube Mining

Searching for interesting patterns in attributed graphs

Dissertation presented by
Florian DEMESMAEKER

for obtaining the Master's degree in
Computer Science and Engineering

Supervisors
Amine GHRAB, Siegfried NIJSSEN

Readers
Pierre DUPONT, Sabri SKHIRI

Academic year 2016-2017

Abstract

Due to the availability of rich network data, graph mining techniques have been improved to handle the emergence of such heterogeneous data. Exploratory data analysis in rich networks aims at discovering interesting information in complex data and is an open challenge. In this work we search for repeating structures or *patterns* that represent an association between two entities according to some of their features. Patterns can be found in social networks. Let us say that we observe a high number of relationships between American and Chinese people. This is a pattern that 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 search for surprising features associations in networks that we model by attributed graphs. To quantify how interesting a pattern is, we use a hypothesis testing framework. We define a *null* model that we assume has generated the data at hand. We then evaluate how likely it is that a pattern occurs under the *null* model.

The earlier mentioned type of pattern can be found using the graph cube framework proposed by Zhao et al. This framework defines one network per combination of node features. Afterwards we show that there is a relationship between our graph mining approach and the frequent itemset mining literature. We propose a mapping from pattern mining in attributed graphs to a frequent itemset mining setting and compare our method with some techniques from this literature. Different experiments are performed on the MovieLens dataset and we show the interpretability of the results.

Keywords - Data mining, pattern mining, hypothesis testing, graph mining, graph cube.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my supervisor Prof. Siegfried Nijssen for his patience, constructive feedback and support throughout the year.

I would also like to sincerely thank my supervisor Amine Ghrab for offering me the chance to work on this project, his support and constructive comments.

Then, I would like to acknowledge the other members of my thesis committee, Sabri Skhiri for his insightful comments during our meetings at Euranova and Prof. Pierre Dupont for his wise advice at the very beginning of my master's thesis.

My sincere thanks go to my parents, my brother and my partner for their constant love and support for all these years.

Last but not least, I would like to thank my friends present in the *David Parnas*' room at the INGI department during these last two years for the friendly and hard-working environment in the room.

Contents

List of Symbols and Abbreviations

Symbol	Meaning
$\mathcal{N}$	Multidimensional network
$\mathcal{L}$	Graph cube lattice
Ω	Null hypothesis
$\vee$	Logical OR
$\wedge$	Logical AND
S	A selection of attributes, also called an aggregation or a cuboid
tid	transaction identifier
$\Pr(X)$	Probability of X
O	Big O complexity
RDB	Relational Database
OLAP	On-Line Analytical Processing
MINI	Mining Informative Non-redundant Itemsets
MTV	Maximally informaTiVe (summary)

List of Figures

List of Tables

Introduction

Due to the increasing availability of network data, graph mining turned into a high topic of interest in the data mining literature. 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 thesis we search for repeating structures or *patterns* that represent an association between two entities according to some of their features. Let us say that we observe a high number of relationships between American and Chinese people. This is a pattern that 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 surprising patterns. These are features associations but not individual edges of the graph. To measure how surprising a pattern is, we define a *null* model that we assume has generated the data at hand.

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 [20] or anomaly mining by modeling a normative pattern [11]. Chakrabarti proposes Autopart [8] 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 [4]. For instance Gao et al. present CODA [13], an algorithm to mine community outlier nodes by considering both node information and the network topology.

The purpose of our miner differs from these approaches by finding surprising patterns that consist of features associations. The earlier mentioned type of pattern can be found using the graph cube framework proposed by Zhao et al. [28]. The graph cube model defines one network per combination of node features.

Moreover we compare our method with MINI [12] and MTV [19], two algorithms from the frequent itemset mining literature. We will show in this work that there is a relationship between mining patterns in attributed graphs and frequent itemset mining. We propose a mapping from attributed graphs to transactional databases in this thesis. The remainder of this thesis is organized as follows.

Chapter 1 presents the graph cube model and illustrates it by a toy example that is reused throughout the thesis.

Chapter 2 presents the frequent itemset mining problem in general, the motivation to use this setting and two algorithms having a different notion of interestingness of a pattern.

In chapter 3 we propose the mapping from an attributed graph to a transactional database.

Then we formally define a pattern within the graph cube data model and propose a corre-

sponding measure of interestingness in chapter 4.

Chapter 5 presents the experiments we conducted on two different MovieLens datasets and how we built attributed graphs from these datasets.

Finally we conclude this thesis and discuss future work in the last chapter.

Chapter 1

The graph cube model

We are interested in mining informative patterns in a graph whose nodes have features. All possible graphs lie in a multidimensional space enhanced by the interactions between data points. Hence we propose to use an On-Line Analytic Processing (OLAP) approach. The aim of OLAP is to summarize the data typically by answering questions such as "How did the average number of cell phone calls per year per person in Belgium change from 10 years ago to today?". These are queries that often involve grouping operations on one or more attributes contained in the data. An OLAP *cube* refers to a multidimensional array of data. Each cell of the cube contains aggregated information. In this thesis we are interested in the links between such cells, more precisely the number of links between two entities. This is why we base our work on the graph cube model presented by Zhao et al [28]. In this paper they present the model on which one can perform OLAP queries on a graph cube instead of a regular cube.

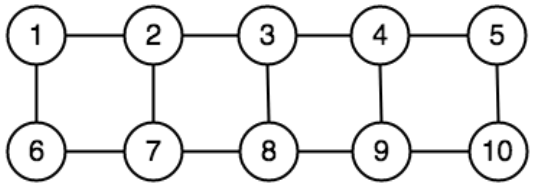
As an example let us consider a network whose nodes represent people with certain demographic attributes and there is an edge between two persons if they called within last month. On a graph cube built upon that network one can ask richer queries than on a data cube such as "What is the network structure between people as grouped by location?". The result of this query consists of a graph built from the original graph whose nodes and edges have been merged. More precisely the obtained graph contains as many nodes as the total number of user locations. This result contains more information than a traditional OLAP data cube query. In the remainder of this chapter we present the graph model on which we base our work. We also define the possible operations on such a data structure. The definitions in the following sections come from [28].

1.1 Multidimensional network

First we introduce the concept of multidimensional network, a graph with attributes on the nodes. This is the data structure on which the graph cube is based. We define it formally in definition 1.1.

Definition 1.1. 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.

Example 1.1 presents a possible instance of such a network. As we will use this example throughout this work we call it our *toy network* and we will refer to it with this name. In chapter 3 we will analyze it in more depth. Please note that several people in the example have the same values for all their attributes. For instance, nodes 3 and 7 are women living in Belgium



(a) Graph

ID	Gender	Location	Profession
1	M	BE	Engineer
2	M	NL	Lawyer
3	F	BE	Teacher
4	M	DE	Engineer
5	M	DE	Lawyer
6	F	NL	Teacher
7	F	BE	Teacher
8	M	NL	Engineer
9	F	DE	Teacher
10	F	DE	Lawyer

(b) Vertex attribute table

Figure 1.1: The toy network

and working as teachers. They can only be distinguished by their identifiers. We could have merged them and put a weight of 2 in the resulting node, as we will see below for aggregate networks.

Example 1.1. We present an example of an aggregate network in figure 1.1. It contains $|V| = 10$ nodes and $|E| = 13$ edges. It is defined by $n = 3$ dimensions: Gender, Location and Profession. To each node is associated a multidimensional tuple to represent its attributes, e.g. $A(1) = (\mathrm{M}, \mathrm{BE}, \mathrm{Engineer})$ if we denote a node $u \in V$ by its ID.

1.2 Aggregate network

The graph cube structure consists of a number of aggregate networks. These are multidimensional networks whose one or several dimensions have been aggregated. The obtention of an aggregate network is similar to the effect of a `GROUP BY` in a relational database: the nodes sharing the same dimension values are merged into a single unit following an aggregation function. To help us define an aggregate network we first introduce the definition of the equivalence of two nodes with respect to a given aggregation.

Definition 1.2. 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 $A'_i = *$ 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.

To illustrate this definition let us consider the toy network presented in example 1.1 and the aggregation $A' = (\text{Gender}, \text{Location}, *)$. Nodes 4 and 5 are equivalent according to A' as they both represent men living in Germany while nodes 6 and 7 are not equivalent according to A' as they both represent women being teachers but not living in the same country.

Similarly we say that two edges are equivalent according to an aggregation A' if they depict the same relationship. That is if the associated nodes of the two edges are equivalent pair by pair. For instance the edges (2)-(7) and (3)-(8) both depict a relationship between a woman living in

Belgium and a man living in the Netherlands. Therefore according to $A' = (\text{Gender}, \text{Location}, *)$ the two edges are equivalent. Please note that since we consider undirected edges the order of the nodes can be permuted when considering an edge. In our example nodes (2) and (8) are equivalent according to A' as well as nodes (7) and (3). The order of the nodes in the pair associated to the edge can be reversed. We give a formal definition of the equivalence between two edges according to an aggregation below.

Definition 1.3. 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

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

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.

The definitions of equivalence 1.2 and 1.3 give an indication on how to group the nodes and the edges of a given network to form an aggregate network. Indeed all nodes equivalent one to another in the original network are merged into a single node in the aggregate network. We are left with the choice of what is the information summarizing the elements of the network. For the sake of simplicity and for our purpose we set this information to the number of summarized elements. As a consequence an aggregate node u gets a weight equal to the number of nodes equivalent to u in the base graph. A similar reasoning applies to the edges as well.

Definition 1.4 formalizes these concepts and uses the term *equivalence set* to denote a set of elements that are equivalent in the sense of definitions 1.2 and 1.3. For instance in example 1.1 and according to aggregation $A' = (\text{Gender}, \text{Location}, *)$ nodes 2 and 8 belong to the same equivalence set. Correspondingly edges (2)-(7) and (3)-(8) belong to the same equivalence set. As a consequence a network may contain several equivalence sets of nodes as well as several equivalence sets of edges.

Definition 1.4. 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

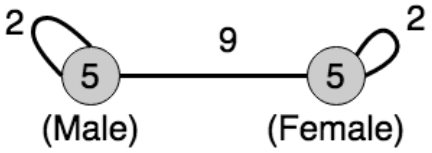
1. To every equivalence set of nodes V_{eq} of V, a node in the aggregate network $v' \in V'$ is associated. The weight of v' is the number of equivalent nodes in V_{eq}, i.e. $w(v') = |V_{eq}|$. Therefore v' is called a condensed vertex.

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.

Considering the network of example 1.1 as a *base*, we can define aggregations such as (Gender, $*$, $*$) and (Gender, Location, $*$). Example 1.2 presents the networks obtained by summarizing on these two aggregations, alongside their corresponding relational tables.

Example 1.2. Figure 1.2a presents an aggregate network derived from the toy network, summarized with respect to the aggregation (Gender, $*$, $*$). It consists of two condensed vertices Male and Female and three condensed edges. As we mentioned above the weight of each node corresponds to the number of individuals in the base network that comply with the dimension values of the condensed vertex, namely Male or Female, ignoring the other dimension values. The edge weight equal to 1 is not shown. On the other hand figure 1.2b presents a traditional group-by along the dimension Gender on the vertex attribute table shown in 1.1b.

Figure 1.3a presents another aggregate network also derived from the base network and summarized with respect to (Gender, Location, $*$). In contrast, figure 1.3b presents a traditional group-by along the dimensions Gender and Location on the vertex attribute table shown in 1.1b.

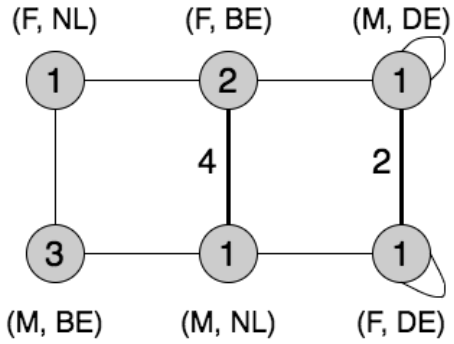


(a) Aggregate network

Gender	COUNT(∗)
Male	5
Female	5

(b) Aggregate table

Figure 1.2: Aggregate network w.r.t (Gender, ∗, ∗) and its corresponding RDB



(a) Aggregate network

Gender	Location	COUNT(∗)
Male	LU	1
Female	LU	2
Female	NL	2
Male	BE	3
Male	DE	1
Female	DE	1

(b) Aggregate table

Figure 1.3: Aggregate network w.r.t (Gender, Location, ∗) and its corresponding RDB

1.3 Graph cube lattice

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. Highly based on [28], definition 1.5 and the following comments explore this question while example 1.3 illustrates the relationships between the networks of the graph cube.

Definition 1.5. Given a multidimensional network $\mathcal{N} = (V, E, A)$, the **graph cube** is obtained by restructuring $\mathcal{N}$ in all possible aggregations of A. For each aggregation A' of A, the measure is an aggregate network G' with respect to A', as defined in definition 1.4.

An aggregation of a multidimensional network $\mathcal{N} = (V, E, A)$ is also called a **cuboid**. The **size** of a cuboid A' of A is $(|V'|+|E'|)$ where V' and E' are the vertex and edge set of the aggregate network corresponding to A' respectively. For a cuboid A', $dim(A')$ denotes the set of non-$*$ dimensions of A'. For instance if $A' = (\text{Gender}, \text{Location}, *)$ then $dim(A) = \{\text{Gender}, \text{Location}\}$.

Let us consider two cuboids B and C. B is an **ancestor** of C if $dim(B) \subseteq dim(C)$ and therefore C is a **descendant** of B. In particular if $|dim(C)| = |dim(B)| + 1$, B is a **parent** of C and C is a **child** of B. For example (Gender, ∗, ∗) is a parent, and thus an ancestor, of (Gender, Location, ∗).

If $dim(B) = dim(C) = l$ then B and C are **siblings**, located at the **level** l of the graph cube. Two cuboids are distinguished:

1. The **base cuboid** A_b with $|dim(A_b)| = n$ is a descendant of all other cuboids of the graph cube.

2. The **apex cuboid** A_{all} with $|dim(A_{all})| = 0$ is an ancestor of all other cuboids in the graph cube.

Let us denote the set of all cuboids of a graph cube as 2^A, the power set of A. We can observe that a partial order exists within the graph cube, based on the set of dimensions $dim(\cdot)$. Therefore a **graph cube lattice** $\mathcal{L} = \left\langle 2^A, \subseteq \right\rangle$ can be induced by the partial ordering $\subseteq$ upon 2^A. Example 1.3 presents the lattice induced by our base multidimensional network of figure 1.1.

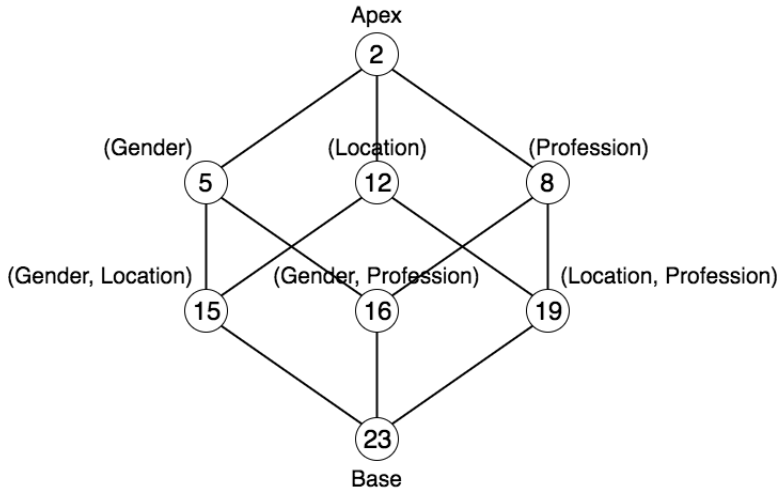


Figure 1.4: The graph cube lattice

Example 1.3. Figure 1.4 presents the graph cube lattice induced by our base network of figure 1.1. Each vertex is a cuboid and each edge represents a parent-child relationship between two cuboids. The size of each cuboid is shown within the corresponding vertex. For each cuboid in the graph cube there is a unique corresponding aggregate network. Let us consider again the two distinguished cuboids $A_b = (\text{Gender}, \text{Location}, \text{Profession})$ and $A_{all} = (*, *, *)$. The aggregate network corresponding to the base A_b is the original multidimensional network while the aggregate network for the apex A_{all} contains a unique vertex, with a self-loop in our case.

The networks corresponding to the aggregations in the graph cube contain information at different detail levels. A descendant of an aggregation A' is linked to a network with a finer-grained view than the network of A'. For instance the network of the cuboid $(\text{Gender}, \text{Location}, *)$ is more detailed than the network of $(\text{Gender}, *, *)$.

Chapter 2

State of the art

In chapter 3 we will propose and demonstrate a relationship between pattern mining in attributed graphs and itemset mining. Therefore it is important to first present the itemset mining setting. This model was introduced to analyze the baskets of people in supermarkets. Each basket corresponds to a transaction and each transaction contains some items.

In this chapter we present the frequent itemset model and different techniques to mine interesting frequent itemsets. We can transform the problem of finding interesting aggregate edges of any network in the graph cube lattice induced by an attributed graph to the problem of finding interesting itemsets in a transactional database. The idea is that each edge in the base 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. Chapter 3 presents and discusses the mapping in details.

2.1 Frequent itemset mining

First we present the market-basket model widely used in the frequent itemset literature in subsection 2.1.1. It was first introduced by Agrawal et al. [1]. Then we discuss how to reduce the number of frequent itemsets that are found and an alternative way to represent a transactional database.

2.1.1 The market-basket model

The frequent itemset mining settings are based on the market-basket model. We are given a set of transactions $\mathcal{T}$ each containing a set of items. The items belong to the set of all items $\mathcal{I}$. A transaction can be seen as the set of items a customer buys at a supermarket. A transaction is often represented by a tuple (tid, I) consisting of a unique identifier and a set of items I. The task of a frequent itemset miner is to discover items that occur together in many transactions. The power set of $\mathcal{I}$ contains all the possible itemsets a miner could consider.

Once the frequent itemsets have been found, the purpose of this kind of miner is to build *association rules*. These rules have the form $I \rightarrow J$. Using the supermarket analogy it means that people who bought itemset I are likely to also buy itemset J.

Definition 2.1. The **support** of an itemset I is the number of transactions containing I, i.e. $supp(I) = |\{(tid, X) \in \mathcal{T} \mid I \subseteq X\}|$.

One parameter of such a miner is the threshold from which we consider an itemset to be frequent, i.e. to occur in enough transactions.

Definition 2.2. The **support threshold** σ of a frequent itemset mining task is the minimum support an itemset must have to be considered as *frequent*.

As the number of itemsets is exponential with the number of items, efficient algorithms are used to mine frequent itemsets. Agrawal and Srikant presented the Apriori algorithm [2] to perform such a task, a first efficient algorithm. It performs a pruning based on the *anti-monotonicity* property of an itemset frequency, which states that if an itemset I is not frequent then none of its supersets is. Example 2.1 presents a small database of transactions and the itemsets that can be found. Furthermore it shows that not every frequent itemset is interesting.

Example 2.1. Table 2.1 presents a database of $|\mathcal{T}| = 5$ transactions and $|\mathcal{I}| = 6$ different items. Setting a support threshold $\theta = 3$ we observe the following frequent itemsets: {Beer}, {Bread}, {Diaper}, {Milk}, {Beer, Diaper}, {Bread, Diaper}, {Bread, Milk}, {Diaper, Milk}. As bread and milk are common needs on an every day basis, their occurrences in the itemsets are not really surprising and therefore not that useful to report. On the other hand the itemset {Beer, Diaper} is quite surprising. Therefore the support of an itemset is not enough to determine its relevance, its importance. We observe here the need for an additional measure to take this into account.

TID	Items
1	Bread, Diaper, Milk, Soap
2	Beer, Bread, Diaper, Eggs
3	Beer, Diaper, Milk, Soap
4	Bread, Milk
5	Beer, Bread, Diaper, Milk

Table 2.1: An example transactional database

2.1.2 Condensed representations of itemsets

The support threshold is a parameter restricting the set of patterns to the frequent itemsets. However a phenomenon arises with frequent itemset mining, the so-called *pattern explosion*. Indeed the anti-monotonicity of the itemset frequency implies that if an itemset I is frequent then so are its subsets. Since the number of subsets of I is exponential with its size a prohibitively large number of frequent itemsets is found. As a consequence several condensed representations of the frequent itemsets have been proposed among which the maximal and the closed [21] itemsets.

An itemset is maximal if none of its supersets is frequent. Therefore one can deduce all the frequent itemsets (FI) from the set of maximal frequent itemsets (MFI). Considering MFI reduces the size of the miner output but looses the frequencies information. To be able to restore the frequencies one should keep the set of closed frequent itemsets (CFI).

Definition 2.3. An itemset is **closed** if none of its immediate supersets has the same support.

Although CFI is smaller than FI, it is larger than MF but one can deduce all the frequent itemsets from it. Let us consider two itemsets $I \subset J$. The anti-monotonicity principle states that $supp(J) \leq supp(I)$. As a consequence if J is closed and I is not, $supp(I) = supp(J)$. Algorithms using constraint programming have been proposed to mine efficiently closed [25] and maximal [6] frequent itemsets.

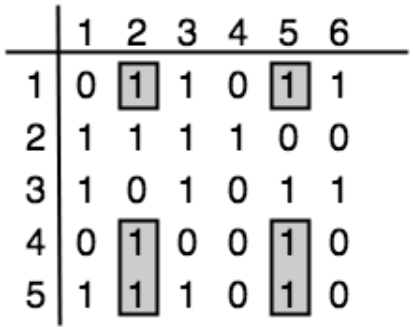


Figure 2.1: A binary database equivalent to table 2.1 with a highlighted tile

2.1.3 Itemsets in a binary database

Under the market-basket model presented above a database is a set of transactions each consisting of a set of items. We can represent the same information as a binary database. This is a n-by-m rectangular matrix D for n transactions and m items. If transaction i contains item j then $D(i, j)$ is set to 1, otherwise it is 0.

Example 2.2. Figure 2.1 presents the binary database representation of table 2.1. Items are numbered by their lexical order. Namely the mapping (Item, number) is the following: (Beer, 1), (Bread, 2), (Diaper, 3), (Eggs, 4), (Milk, 5), (Soap, 6).

This representation can be used to mine tiles [14] in such a database. A tile is a block of ones not necessarily contiguous but they must form a rectangle full of ones. Figure 2.1 highlights a tile present in our database example. A tile correspond to an itemset and its support. For instance the highlighted tile corresponds to the itemset {Bread, Milk} as it spans columns 2 and 5. Furthermore this itemset has a support of 3 as the tile spans three rows, that is three transactions. Several approaches in the literature mine a set of potentially overlapping tiles that are informative, e.g. [10, 18]. Such a set of tiles is equivalent to a set of itemsets. In section 2.3.2 we present an approach that aims at summarizing a database by mining tiles that are informative with respect to the maximum entropy principle.

2.2 Interesting itemsets

As mentioned in section 2.1, if an itemset is frequent it does not mean that it is interesting. Gallo et al. present MINI [12], an algorithm reporting the Most Informative and Non-redundant Itemsets. This algorithm takes as input the set of closed frequent itemsets of a database. The reported itemsets are qualified as informative because they are surprising with respect to some prior information on the database. Moreover MINI ensures that overlapping itemsets are not both highly ranked as informative as they convey part of the same information.

2.2.1 The interestingness measure

Let us denote by D the data we have at disposal and $\mathcal{D}$ the space from which D is drawn. The idea behind MINI is to select itemsets that are hard to be explain by chance under prior knowledge on the data $D \in \mathcal{D}$. This prior information is formulated as a *null hypothesis*, a set of assumption(s) about the distribution P_D generating the data $D \in \mathcal{D}$. Although it is possible that several distributions respect the assumptions, we consider that there is a unique P_D for conciseness.

Here in the itemset settings MINI considers two null hypotheses Ω_i and Ω_t, the first about the frequencies of the items and the second about the frequencies of the transactions. Both hypotheses

follow an independence model. Under Ω_i the model depicts the "uninteresting" situation where no association between the items is present. That is each item occurs independently from the others. We can model the presence of an item i to appear in any particular transaction as a Bernoulli random variable. Its presence probability is equal to its frequency, i.e. $p_i = \frac{supp(i)}{n}$ where $n = |\mathcal{T}|$. Then under Ω_i the probability of an itemset I to be included in a random transaction is given by equation 2.1. The second assumption Ω_t is similar to Ω_i and will be discussed below.

$$p_I = \prod_{i \in I} p_i \tag{2.1}$$

Taking back the example transactional database represented in table 2.1 let I and J denote the itemsets {Bread, Milk} and {Beer, Diaper} respectively. The probability of I is equal to the product of the proportion of transactions containing Bread and the proportion of transactions containing Milk. Hence $p_I = 0.8 \times 0.8 = 0.64$ is the probability to observe I under the independence model.

Silverstein et al. [24] defined the interest of an itemset I to be the ratio between the support of I and the expected support of I under the independence model. The expected support of I is given by np_I with n the number of transactions in the database. In our example, the expected support of I={Bread, Milk} is 3.2. On the other hand the actual support of I is 3. Hence its interest is equal to 0.94 which means that it is expected to appear a bit more than it actually does. Using a similar reasoning $p_J = 0.48$ and the expected support of J is 2.4, in contrast with an actual support of 3. Consequently the interest of J is equal to 1.25 which means that it appears in more transactions than expected under the independence model. The idea of MINI is also to measure the interest of an itemset but using a more sophisticated approach.

Each itemset I is assigned a *p-value* that describes the probability of I to be surprising. An itemset I is surprising if its support is greater or equal to the support it would have under the independence assumption Ω_i. Formally p-value$(I) = \Pr(supp_{\Omega_i}(I) \geq supp(I))$ where $supp_{\Omega_i}$ is the support of I under Ω_i and $supp(I)$ is the actual support of I. The lower the p-value of I the lower the probability for $supp(I)$ to be explained by chance under the null hypothesis. Hence Gallo et al. suggest to use the p-value as a measure of interestingness, the smaller the p-value the more informative I is. The remainder of this section presents the details to calculate such a p-value.

Under Ω_i the probability that the support of an itemset I is exactly $supp(I)$ under the independence model is given by the means of the binomial distribution, as stated in equation 2.2.

$$\Pr(supp(I)) = \binom{n}{supp(I)} p_I^{supp(I)} (1 - p_I)^{supp(I)} \tag{2.2}$$

The probability of observing $supp(I)$ for an itemset I can be seen as trying to observe I in each transaction consecutively, each trial having a probability of success p_I.

The p-value for an itemset I under the independence model is defined as the probability of having a support greater than or equal to $supp(I)$. This is the probability to observe a surprising itemset I. It is given by the means of the cumulative binomial distribution function, as stated in equation 2.3.

$$P_c^I = \sum_{s=supp(I)}^{n} \binom{n}{s} p_I^s (1 - p_I)^{n-s} \tag{2.3}$$

The p-value for an itemset I is then denoted by P_c^I under this first model.

A second model can be built in the frequent itemset settings. One can consider a model where the transactions are independent from each other, i.e. any transaction contains itemset I with equal transaction-dependent probability $f_t = \frac{supp(t)}{m}$ where $supp(t)$ is the number of items in t and $m = |\mathcal{I}|$. Our pattern of interest under this model is a *set* of transactions T. The support of T is equal to $supp(T) = |\{i \mid \forall t = (tid, X) \in T, i \in X\}|$, that is the number of items shared by the transactions in T.

Then following a similar reasoning as above we consider under Ω_t the probability that a set T is surprising, i.e. its p-value. It is given by P_c^T in equation 2.4.

$$P_c^T = \sum_{s=supp(T)}^{m} \binom{m}{s} p_T^s (1 - p_T)^{m-s}, \quad \text{with} \quad \prod_{t \in T} f_t \tag{2.4}$$

As an itemset I is assigned two p-values from two different independence models, we should make a choice on the final p-value of I. The larger the p-value the lesser surprising, the more consistent with the independence model I is. The maximum between the two p-values should be taken as the actual p-value of I because it should be the one that best explains the data. In other words the p-value of I should not overestimate how interesting I is. To achieve this we use all the knowledge we have at disposal. Therefore equation 2.5 defines the p-value of an itemset.

$$\text{p-value}(I) = \max(P_c^I, P_c^T) \tag{2.5}$$

The p-value of an itemset I will be small if its items are more strongly associated than expected. This fact is induced by the two assumptions Ω_i and Ω_t. On the one hand itemsets that are larger and that have a higher support will tend to have a smaller p-value because $P_I = \prod_{i \in I} p_i$ will decrease and the starting value s of the sum in 2.3 will increase. On the other hand p-value will be smaller also if the individual frequencies of the items in I are smaller because P_I will decrease.

Besides based on the fact that the support of all the itemsets can be derived from the support of the closed itemsets, Gallo et al. present theorem 2.1.

Theorem 2.1. *For any non-closed itemset there exists a closed itemset that has a lower p-value, i.e. that is more significant.*

This theorem allows MINI to restrict its input to the set of closed itemsets of a database. Several algorithms have been proposed to mine closed itemsets among which CHARM [27] and LCM [25].

2.2.2 The MINI algorithm

Gallo et al. present MINI, an algorithm for mining informative non-redundant itemsets. It takes as input the list of closed itemsets $\mathcal{C}$ of a database. First the itemsets in $\mathcal{C}$ are sorted by increasing p-value. After that an iterative procedure takes place. The p-values of the list are updated in order to ignore itemsets that are redundant with highly ranked ones. The itemsets overlapping with highly listed ones are penalized by incrementing their p-value so that they go down in the list.

Let us describe an update step. Suppose I_k is the itemset at position k that is going to be updated. Table 2.2 illustrates the context of the update. All the itemsets $\mathcal{I}_{k-1} = \bigcup_{l=1}^{k-1} I_l$ are more highly ranked than I_k, they are said to be *covered*.

Considering the basic case of a single itemset being covered, i.e. I_2 is going to be updated and thus $\mathcal{I}_1 = I_1$. The null distribution is modified after the update, reflecting the fact that I_1 is covered. This new distribution still models a random transaction and is defined using a mixture model that consists of two components c_1 and c_2. The first one corresponding to the initial null

I_1
I_2
...
I_k
...
$I_{\lvert\mathcal{C}\rvert-1}$
$I_{\lvert\mathcal{C}\rvert}$

$\mathcal{I}_{k-1}$

Increasing p-value

Table 2.2: The sorted list of closed itemsets considered by MINI

hypothesis Ω_i and the second one modeling a transaction that supports all items in I_1. Therefore the prior probabilities of a transaction being respectively in c_1 and c_2 are given by equations 2.6 and 2.7.

$$\Pr(c_1) = 1 - \frac{supp(I)}{n} \tag{2.6}$$

$$\Pr(c_2) = \frac{supp(I)}{n} \tag{2.7}$$

The conditional probability that a random transaction $\mathtt{t}$ supports a given itemset I for each of the components is given by equations 2.8 and 2.9. The set $is(\mathtt{t})$ contains all the items included in transaction $\mathtt{t}$.

$$\Pr(I \in is(\mathtt{t}) \mid c_1) = \prod_{i \in I} p_i \tag{2.8}$$

$$\Pr(I \in is(\mathtt{t}) \mid c_2) = \prod_{i \in I \setminus I_1} p_i \tag{2.9}$$

In component c_2 a random transaction always includes the items in I_1. Consequently the product of probabilities p_i in equation 2.9 does not consider the items $i \in I_1$.

By combining the prior and the conditional probabilities we obtain the probability of an itemset I to be supported by a random transaction $\mathtt{t}$ under the new null distribution. It is formally given in equation 2.10. The superscript (1) denotes the update of the probability of I given that I_1 is covered.

$$p_I^{(1)} = \Pr(I \in is(\mathtt{t})) = \left(1 - \frac{supp(I)}{n}\right) \prod_{i \in I} p_i + \frac{supp(I)}{n} \prod_{i \in I \setminus I_1} p_i \tag{2.10}$$

If I and I_1 are disjoint then the probability $p_I^{(1)}$ is equal to the previous probability p_I as given in equation 2.1. On the opposite the larger the overlap between I and I_1 the fewer factors p_i smaller than 1 in the second product of equation 2.10. Then the p-value of itemset I is recomputed and has increased. Indeed the higher p_I in the equation of the p-value 2.3 the higher the p-value P_c^I.

Equation 2.10 can be generalized [12] to take into account several covered itemsets in $\mathcal{I}_{k-1}$ when updating itemset I_k. The MINI algorithm traverses the ordered list as shown by table 2.2. The first itemset I_1 is covered by default since there is no more informative itemset than I_1. Then the second itemset I_2 is considered and its p-value is updated.

Gallo et al. state and prove that the updated p-value of an itemset I regarding a set of h covered itemsets denoted by p-value$^{(h)}(I)$ increases with h. Therefore a greedy strategy is

adopted. Following the update of the p-value of I_1, if p-value$^{(1)}(I_2)$ is smaller than the current p-value of the subsequent itemset I_3 then I_2 is covered and the process continues. But if p-value$^{(1)}(I_2) \geq$ p-value(I_3) then I_2 is inserted in the list of closed itemsets such that the order is preserved. As a consequence I_3 becomes the second itemset in the list and is considered next. This process continues until a user specified number of itemsets are covered.

As MINI is a greedy algorithm it is not complete. For instance it might output that an itemset is informative while it can be explained by other itemsets. Let us illustrate this situation with an example. If the items A and B are strongly correlated, C and D are strongly correlated, but $\{A, B\}$ and $\{C, D\}$ are independent, then MINI will report the two correlations. Nevertheless if the p-value of itemset $\{A, B, C, D\}$ is smaller than the p-values of $\{A, B\}$ and $\{C, D\}$, the larger itemset will be selected first despite the fact that reporting $\{A, B\}$ and $\{C, D\}$ separately is a more refined piece of information.

2.3 Maximum entropy models and subjective interestingness

This section presents a family of models inspired by the maximum entropy principle [17] to describe prior information one can have on some data. Section 2.3.1 presents the entropy measure which is used throughout this section. In section 2.3.2 we present a generic framework that aims to encompass the earlier mentioned family of models. This generic framework is an introduction to the MTV algorithm [19] whose goal is to summarize a transactional database with a set of itemsets. This approach is different from the goal of our graph miner because MTV is looking for representative patterns while we are searching for surprising patterns. However, having a set of representative patterns induces that patterns not present in this set may be surprising.

The maximum entropy principle states that given some stated prior data, the knowledge we have at our disposal is best represented by the probability distribution P_D with largest entropy. The idea is that the selected distribution should admit the most ignorance beyond the stated prior data. Indeed if P_D represents well the prior information then maximizing its entropy on top of that yields a distribution with the most uncertainty beyond the prior information. In other words, based on some prior information, P_D assumes the least knowledge about the data.

Once the prior information has been used to build a probability distribution P_D on the data at hand, it is possible to evaluate the interestingness of a pattern, e.g. by testing the null hypothesis modeled by P_D against the actual support of the pattern. A measure is said to be *subjective* (e.g. [15]) if it does not only depend on the data but also on the prior information one can have on the data. In this section we study some subjective measures of interestingness.

2.3.1 Entropy

The entropy in the sense of Shannon [23] 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 an information. For instance the experiment of tossing a coin needs 1 bit to be encoded since its result is either head or tail. If the coin is fair, the entropy of such a random variable is 1. In general a variable that can take n values requires $\log_2(n)$ bits to be encoded. Formally the entropy $H(X)$ of a discrete random variable X is given in equation 2.11. In the sum the value $\Pr(x_i)$ is the probability to observe the i-th event of the variable X.

$$H(X) = -\sum_{i=1}^{n} \Pr(x_i) \log(\Pr(x_i)) \tag{2.11}$$

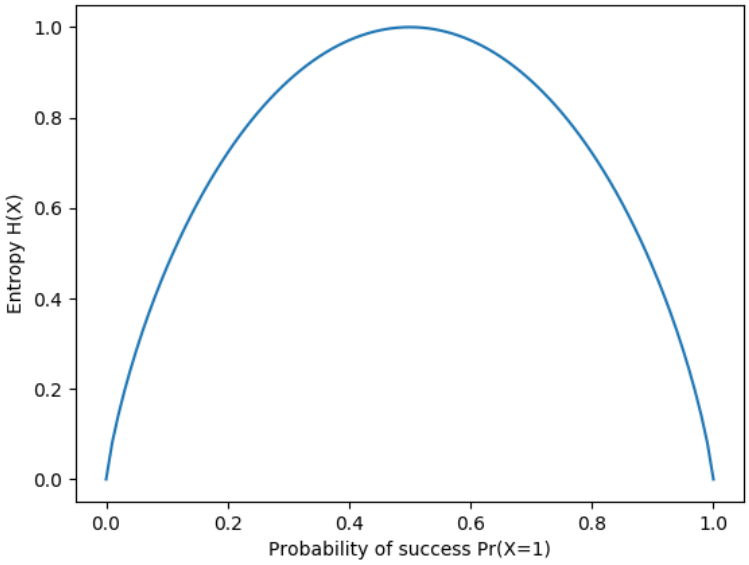


Figure 2.2: The evolution of the entropy of a biased coin

Example 2.3. Let us consider a biased coin with a certain probability of showing a head p. The random variable X can take two values: head or tail. Figure 2.2 presents the evolution of the entropy of X with respect to probability of success p. The entropy is maximum at $p = 0.5$. Indeed the uncertainty is maximal in this setting, no outcome is favored. In the other cases either head or tail is the advantaged result, leading to a higher reduction of information as the more biased the coin becomes.

2.3.2 The general approach

De Bie presented a generic framework to represent prior information using the maximum entropy principle [10] by a probability distribution called MaxEnt. This contribution is a methodology to compute *explicitly representable* probabilistic models for general types of data. The goal is to include a broad class of prior information in a single framework. Then the idea is to quantify the subjective interestingness of a pattern by comparing it with the MaxEnt distribution.

The problem is stated as follows. Find a probability distribution P over the data $\boldsymbol{x} \in \mathcal{X}$ that satisfies a set of linear constraints implied by prior information. The constraints are of the following form

$$\sum_{\boldsymbol{x}} P(\boldsymbol{x}) f_i(\boldsymbol{x}) = d_i \tag{2.12}$$

with f_i real-valued functions of the data and d_i arbitrary values modeling the prior knowledge on the data. Several distributions could satisfy these constraints. But we are looking for the one that makes the least assumption beyond the stated prior data, therefore we search for the distribution that has the largest entropy. Formally the problem is described by

$$\max_{P(\boldsymbol{x})} -\sum_{\boldsymbol{x}} P(\boldsymbol{x}) \log P(\boldsymbol{x}) \tag{2.13}$$

$$\text{such that } \forall i \quad \sum_{\boldsymbol{x}} P(\boldsymbol{x}) f_i(\boldsymbol{x}) = d_i \tag{2.14}$$

$$\sum_{\boldsymbol{x}} P(\boldsymbol{x}) = 1 \tag{2.15}$$

This constrained optimization problem is convex and therefore can be solved using techniques from convex optimization theory [5].

Example 2.4. As an instantiation of the general procedure described above we consider a binary database D containing n rows and m columns. Each entry of D is binary, i.e. $\forall i, j \quad D(i,j) \in \{0,1\}$. As we mentioned in subsection 2.1.3, D is equivalent to a set of transactions, the pattern of interest is a tile. The prior knowledge considered here consist of the *expected* rows and columns sums. The linear constraints are formalized below.

$$\sum_{D \in \mathcal{D}^{n \times m}} P(D) \left(\sum_j D(i,j) \right) = d_i^r \tag{2.16}$$

$$\sum_{D \in \mathcal{D}^{n \times m}} P(D) \left(\sum_i D(i,j) \right) = d_j^c \tag{2.17}$$

The MaxEnt model can be used to quantify the interestingness of a pattern, for instance by considering its probability under MaxEnt. The smaller this probability the more surprising the pattern. The negative log-probability is known as the self-information [9] in Shannon's information theory, the larger the more informative. Another approach considers the probability of observing the pattern or a stronger occurrence of it also by taking MaxEnt as null hypothesis, this probability is the p-value of a pattern. This is the approach chosen by MINI to quantify the interestingness of an itemset that we developed in section 2.2.

2.3.3 Succinctly summarizing data with itemsets

Mampaey et al. presented the MTV algorithm [19] to summarize data with itemsets. Their work is based on the generic framework of De Bie [10] presented in section 2.3.2 to model the prior information using the maximum entropy principle. As the application is on itemsets we first define the entropy of their distribution p over $\mathcal{T}$. It is given by

$$H(p) = -\sum_{t \in \mathcal{T}} p(A = t) \log p(A = t)$$

where A is the set of all items and $\mathcal{T}$ the set of transactions. The prior knowledge considered in this work is a set of itemsets $\mathcal{C} = \{X_1, \ldots, X_k\}$ that should describe the data well. The constraints modeling the prior information consist of setting the probability of an itemset $p(X_i)$ to its actual frequency $f(X_i)$ in the data. Therefore the set of probability distributions satisfying the constraints on the itemsets in $\mathcal{C}$ is given by

$$\mathcal{P} = \{p \mid p(X_i = 1) = f(X_i), i = 1, \ldots, k\}$$

Those are the distributions we can consider. Then following the maximum entropy principle the proper distribution $p^*_{\mathcal{C}}$ is the one with the highest entropy as given by equation 2.18. Please note that $p^*_{\mathcal{C}}$ is constrained by a set of itemsets $\mathcal{C}$ and then its entropy is maximized.

$$p^*_{\mathcal{C}} = \arg\max_p \{H(p) \mid p \in \mathcal{P}\} \tag{2.18}$$

Once the best distribution $p^*_{\mathcal{C}}$ has been found for a set of itemsets $\mathcal{C}$ one could wonder how to assess the quality of such a model. Mampaey et al. propose to use the Bayesian Information Criterion (BIC) [22] that favors models with fewer parameters. The score of a set of itemsets $\mathcal{C}$ is given by equation 2.19. The smaller the score the better the model.

$$s(\mathcal{C}) = -\log p^*_{\mathcal{C}}(D) + \frac{1}{2}|C| \log |D| \tag{2.19}$$

This score is equal to the sum of negative log-likelihood of the model and a penalty term on the number of itemsets taken into account in the model. As a consequence the better the balance between high likelihood and low complexity of a model the better it is. Such a penalty on the number of parameters is needed because otherwise taking the set of all itemsets would provide the best likelihood. This penalty term ensures that the redundancy of the set of itemsets that best summarize the data is lowered as much as possible.

Given a set of itemsets $\mathcal{C}$ we can compute the maximum entropy model and its BIC score. But the goal is to find the set $\mathcal{C}$ that best summarizes the data, i.e. with the smallest score. Mampaey et al. start from an empty collection and use a greedy approach to select itemset candidates to add to the current collection.

Chapter 3

From attributed graph mining to frequent itemset mining

In chapter 4 we propose a technique to rank patterns in an attributed graph. A possible baseline to compare it with is the MINI algorithm [12] presented in section 2.2.2 which uses a frequent itemset mining setting. This is why in this chapter we propose a procedure to transform an attributed graph into a database that follows the market-basket model, i.e. into a transactional database.

The formal definition of a pattern in an attributed graph is given in chapter 4. However it is important to have in mind the kind of patterns we are looking at in an attributed graph in order to better understand why we establish the mapping given in section 3.1. We are interested in associations between pairs of entities having some characteristics or features. Let $\mathcal{N} = (V, E, A)$ be a multidimensional network as defined in chapter 1. Each node of $\mathcal{N}$ is characterized by $|A|$ attributes, e.g. the location and the gender of a person if the nodes represent people. A pattern is an association between two groups of nodes where all nodes of a group share the same attribute values.

As an example let us consider that the nodes represent people and that each person is characterized by its gender, location and profession. In such a network we may find that the number of connections between men living in the Netherlands and women living in Belgium is surprisingly high. This example pattern is denoted by (M, NL)-(F, BE). It is not an edge of the original network $\mathcal{N}$ but an edge in an aggregate network built from $\mathcal{N}$. The idea behind the mapping we propose next is that each association between a pair of attribute values can be transformed into an item. For instance our example pattern can be represented by an itemset $\{(\mathrm{M}, \mathrm{F}), (\mathrm{NL}, \mathrm{BE})\}$. Hence the edges of $\mathcal{N}$ should be translated into a set of transactions such that a miner finds itemsets representing associations between entities.

3.1 Mapping definition

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. Thus e can be described by a set of pairs of attributes t. As our goal is to use a transaction to describe e the set t can in fact be interpreted as a transaction. Consequently an item consists of a pair of attribute values.

Let us consider an edge $e \in E$ whose end nodes are $u, v \in V$. For any attribute $A_i \in A$ we define an item based on the attribute values of u and v. Let t be the transaction that is going to be created and corresponding to the original edge e. Let X_i be the item corresponding to the

pair of attribute values $A_i(u)$ and $A_i(v)$. Then X_i is equal to the pair $(A_i(u), A_i(v))$.

As an example let the attribute values of u and v be $A(u) = (\mathrm{M}, \mathrm{NL}, \mathrm{Tea})$ and $A(v) = (\mathrm{F}, \mathrm{NL}, \mathrm{Eng})$. Then the transaction t corresponding to the edge associated to u and v consists of three items: $t = \{X_1, X_2, X_3\} = \{(\mathrm{F}, \mathrm{M}), (\mathrm{NL}, \mathrm{NL}), (\mathrm{Tea}, \mathrm{Eng})\}$.

The translation of an edge to a transaction implicitly defines an order between the two nodes. The features of node u are always taken into account before the features of node v in the item they correspond to. But $\mathcal{N}$ contains undirected edges only. Therefore no order between the end nodes of an edge should be considered. Indeed the proposed mapping should add two transactions per edge, one in each direction. Hence in our previous example a second transaction t_2 is added to the database: $t_2 = \{(\mathrm{M}, \mathrm{F}), (\mathrm{NL}, \mathrm{NL}), (\mathrm{Eng}, \mathrm{Tea})\}$. Definition 3.1 formalizes the mapping and example 3.1 shows the induced mapping on a small network.

Definition 3.1. Let $\mathcal{N} = (V, E, A)$ be a multidimensional network and $u, v \in V, e \in E : e = (u, v)$ be two nodes linked by an edge in $\mathcal{N}$. Let $\mathcal{A}$ be the set of all possible node attribute tuples. Then $A(u), A(v) \in \mathcal{A}$. 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 $X_i = (A_i(u), A_i(v)), Y_i = (A_i(v), A_i(u)) \quad \forall i$

Example 3.1. Figure 3.1 presents our toy network from chapter 1 with the attribute values next to the nodes and a label next to each edge. Table 3.1 presents the mapping between the edges of the network and their corresponding pairs of transactions. Each pair of attribute values denoting an item is written without its parentheses and the professions are represented by their first letter only for the sake of clarity. This network contains a lot of relations between people living in Belgium and people living in the Netherlands but no relation between two people living both in Belgium or living both in the Netherlands. Also there are 4 pairs both living in Germany while there are only two pairs of one person living in Germany and the other person living either in Belgium or in the Netherlands. Thus we have a small cluster of people living in Germany but no cluster for the two other countries. Considering the other attributes does not give a remarkable pattern.

This example illustrates the need of two transactions per edge instead of a single one. If edges e_1 and e_3 are solely represented by {MF, BENL, EngTea} and {MF, NLBE, LawEng} respectively, then we cannot infer that both edges represent a relationship between a person living in Belgium and another person living in the Netherlands. Even by inverting the order of the tuples in the mapping that output e_3, namely by converting e_3 to {FM, BENL, EngLaw}, we still removed a piece of information. Indeed we cannot infer anymore that e_1 and e_3 represent a relationship between a man and a woman. As a consequence the mapping we defined above yields a transactional database that is more consistent with the original attributed graph since it maps two transactions to a single edge.

3.2 Itemsets and graph patterns

We proposed in the previous section a mapping from an attributed graph to a transactional database in order to compare frequent itemset mining algorithms with our graph mining algorithm presented in chapter 4. In this section we discuss the kind of patterns that are found in a transactional database, especially under the independence model. We use the graph transformed

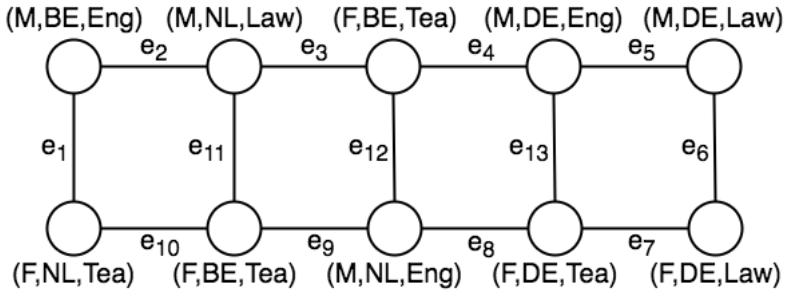


Figure 3.1: The toy network with attribute values

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 3.1: Mapping from the toy network to its corresponding transactional database

into a transactional database in example 3.1 throughout the section as an illustration.

Let $\mathcal{N}$ be the network from which we built the transactional database D. Although D is a traditional set of transactions we should note that it has a certain property. As an edge is mapped to two transactions there is some redundancy in the database. Indeed each time an item $X_i = (v_1, v_2)$ representing a relation between two attribute values v_1 and v_2 is present in a transaction, then its reversed version (v_2, v_1) is present in another one. For instance in table 3.1 there are as many (F, M) items as (M, F) items. But in the particular case where item X_i represents a relation between to identical attribute values, i.e. $X_i = (v_1, v_1)$, it means that the support of X_i in the database is doubled. Therefore in D we have a bias that consists of giving a higher frequency to items representing a self-loop relationship.

3.2.1 Interesting itemsets

We presented the simple measure of Silverstein et al. [24] to quantify the *interest* of an itemset in section 2.2.1 and the MINI algorithm [12] that finds informative and non-redundant itemsets in section 2.2.2. Considering an itemset I the higher the ratio $\frac{supp(I)}{supp_{\Omega_i}(I)}$ the higher the interest of I where $supp_{\Omega_i}(I)$ is the expected support of I under the independence model. On the other hand the lower the p-value of I the more informative I is according to MINI. The p-value is equal to $\Pr(supp(I) \leq supp_{\Omega_i}(I))$. The smaller the p-value the less likely it is to observe a support lower than the expected support. Hence a surprising pattern denotes a pattern whose support is

Itemset	Interest
{FM, DENL, TeaEng}	16.67
{MF, NLDE, EngTea}	16.67
{FM, BEDE, TeaEng}	16.67
{MF, DEBE, EngTea}	16.67
{FF, BENL, TeaTea}	10
{FF, NLBE, TeaTea}	10
{MM, BENL, EngLaw}	10
{MM, NLBE, LawEng}	10
{FF, DEDE, TeaLaw}	8.33
{FF, DEDE, LawTea}	8.33

(a) The 10 most *interesting* patterns according to Silverstein's measure

Itemset	p-value
{MF, EngTea}	0.16
{FM, TeaEng}	0.16
{FF, TeaTea}	0.24
{MM, LawEng}	0.24
{MF, NLBE, LawTea}	0.3
{FM, BENL, TeaLaw}	0.3
{MM, EngLaw}	0.32
{DEDE, LawLaw}	0.41
{FF, DEDE, LawTea}	0.79
{FF, DEDE, TeaLaw}	0.79

(b) The 10 most *informative* and non-redundant patterns reported by MINI

Table 3.2: The most surprising patterns of the toy network regarding their *interest* or their quantity of *information*

surprisingly high. We report in tables 3.2a and 3.2b the most surprising patterns according to their *interest* and the most surprising patterns reported by MINI respectively.

Let us first note that the 10 most interesting itemsets in table 3.2a corresponds to 5 edges in the original network. Indeed each reported relation appears two times: one in each direction. This phenomenon is a direct consequence of the way we built our transactional database where to each edge corresponds two transactions. The support of an itemset I is always equal to the support of its "reversed" version. For instance the support of {FM, DENL} is the same as the support of {MF, NLDE}. Also the expected support of I depends on the individual supports of the items in I. Those are itemsets of size 1 and hence have the property mentioned above. As the *interest* measure only involves the support and the expected support quantities an itemset I and its reversed version have the same *interest*.

Furthermore we observe that the two most interesting patterns are the two edges between the cluster with interactions between Belgians and Dutchmen and the cluster with German people. As the relations (NL)-(DE) and (BE)-(DE) occur a single time their probabilities are both low and so are their expected supports. These two bridges between clusters should indeed be marked as surprising since most of the relationships in this network are (NL)-(BE) and (DE)-(DE). However it is more informative to report the interaction (NL)-(DE) as surprising than the fully described relation (M, NL, Eng)-(F, DE, Tea) because this is the location information that is surprising here. Knowing that there is only one relationship between a German and a Dutchman it is not surprising that there is only one edge of the form (M, NL, Eng)-(F, DE, Tea). The graph miner we propose in chapter 4 takes this into account.

Regarding the itemsets reported by MINI in table 3.2b we note that $I_4 = \{\text{MM, LawEng}\}$ has a lower p-value than its reversed version $I_7 = \{\text{MM, EngLaw}\}$. Indeed MINI first adds I_4 to its set of covered itemsets and then decreases the p-value of I_7 in order to reduce the redundancy of the reported patterns because the two itemsets share the item MM. The most surprising reported pattern is (M, Eng)-(F, Tea). It is indeed informative as all teachers are women and all engineers are men in this toy example. We could have expected that among the teachers some of them are women and some of them are men. This is what the independence model assumes. Nevertheless the association (BE)-(NL) is not reported as interesting by MINI. The algorithm has a bias towards larger itemsets because they are less likely under the independence model.

Itemset	Score
{FF, DEDE, TeaLaw}	0.04
{FF, DEDE, LawTea}	0.04
{MF, NLBE, LawTea}	0.08
{FM, BENL, TeaLaw}	0.08
{FF, TeaTea}	0.08
{MM, LawEng}	0.08
{MM, EngLaw}	0.08
{DEDE, LawLaw}	0.08
{FM, TeaEng}	0.23
{MF, EngTea}	0.23

Table 3.3: The set of 10 itemsets that best *summarizes* the toy network according to MTV

Finally neither approach reports itemsets of size 1. More particularly in our example the relation (NL)-(BE) is quite interesting because Belgians and Dutchmen do not interact within their own respective country but it is not reported. Our graph mining algorithm is based on an independence model and considers all possible relations as candidates. We will present the most surprising patterns according to our miner in chapter 4.

3.2.2 A set of itemsets as summary

The MTV algorithm [19] that we presented in section 2.3.3 searches for the set of itemsets that summarizes the data best. In the context of our mapping, MTV reports the set of features associations that forms the best summary of the network. We report in table 3.3 the best summary of size 10. The score of an itemset is used by MTV during its greedy search, we reported the final score of each itemset present in the summary. The smaller the more likely the itemset is added to the summary.

We observe that this set contains exactly the 10 most informative patterns reported by MINI that are showed in table 3.2b. The relation (M, Eng)-(F, Tea) is the least interesting itemset but has the highest score while the relation (F, DE, Tea)-(F, DE, Law) is the most interesting itemset but has the smallest score. This situation illustrates the difference between a representative and a surprising itemset. However the other interesting itemsets all belong to the summary. This illustrates the relationship between the two techniques. These are surprising pattern having a p-value small enough to be marked as surprising. The smaller the p-value of an itemset I the more likely it is that its support is surprisingly high. The reported informative patterns having a p-value between 0.24 and 0.41 are convey more information than expected but are not highly anomalous either. Hence they are reported by MTV in the summary.

Chapter 4

Graph mining

In chapter 1 we presented the graph cube model on which we base our work. The problem we are interested in is a data mining task. The present chapter defines the kind of patterns we are looking for in the data as well as some measures related to these patterns. In this thesis we consider that a pattern is an edge of any graph in the graph cube lattice as stated in definition 4.1 and illustrated in example 4.1. Then we define the interestingness of a pattern P to be related to the number of times we *expect* to observe P in the data and the *actual* number of times we observe P in the data.

Problem statement We are given a multidimensional network $\mathcal{N} = (V, E, A)$ as defined in chapter 1. A graph cube lattice is induced from $\mathcal{N}$, containing $2^{|A|}$ cuboids and as many corresponding graphs. We are looking for aggregate edges in any of these graphs that are surprising with respect to some prior knowledge we have on the data.

Section 4.1 formally defines the patterns we are looking at and presents some of them that may be found in the toy network. In section 4.2 we discuss different approaches to model the prior knowledge of a pattern in a network using the ancestors of this network. Then in section 4.3 we compare these approaches on the toy network. We conclude this chapter by presenting the algorithm that searches for surprising aggregate edges.

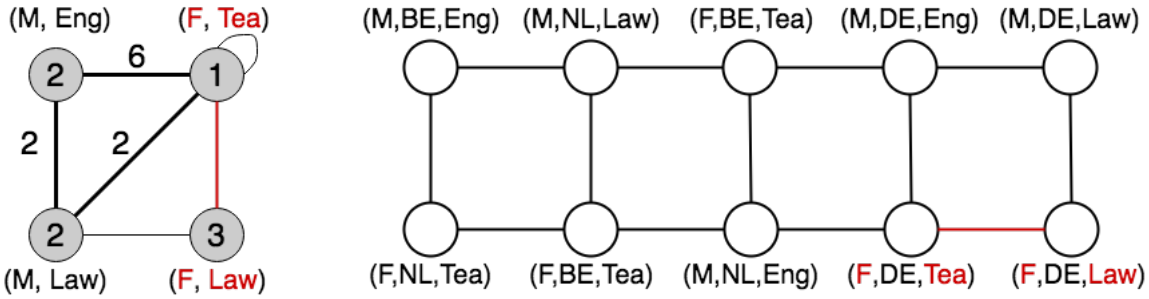
4.1 Pattern definition

Since we use the ancestors of the cuboid where the pattern lies, 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 4.1. 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 4.1. Considering a multidimensional network such as the toy network presented in example 1.1, we can define as many patterns as the total number of edges in all the graphs of the graph cube lattice. Indeed, each aggregate network contains a different combination of features associations. As the two following example patterns will show, a pattern can represent a single edge or multiple ones.

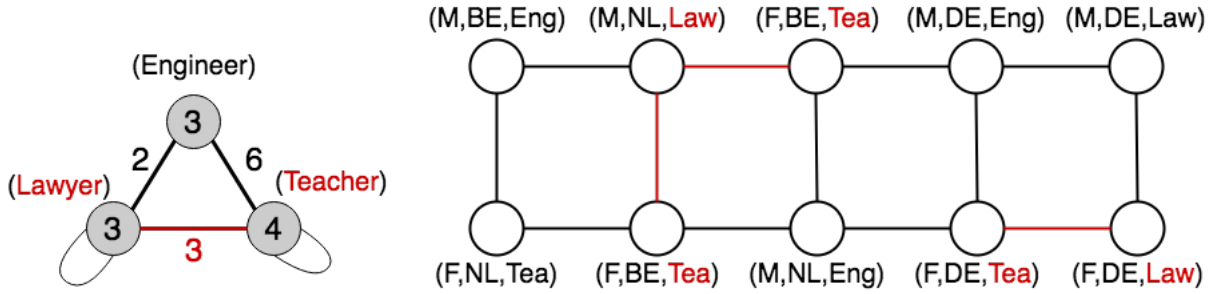
Figure 4.1 presents the pattern (Female, Lawyer)-(Female, Teacher) denoted by P_1 which is present a single time in the data. P_1 is defined upon the cuboid or subset of attributes $S_1 = (\text{Gender}, *, \text{Occupation})$. Figure 4.1a shows P_1 in the aggregate graph corresponding to S_1 while figure 4.1b shows P_1 in the base graph.



(a) The pattern in the aggregate graph

(b) The pattern in the base graph

Figure 4.1: The (Female, Lawyer)-(Female, Teacher) pattern highlighted in the toy network



(a) The pattern in the aggregate graph

(b) The pattern in the base graph

Figure 4.2: The (Lawyer)-(Teacher) pattern highlighted in the toy network

Another pattern denoted by P_2 is presented in figure 4.2: (Lawyer)-(Teacher). P_2 is present 3 times in the data and it is defined upon the cuboid $S_2 = (*, *, \text{Occupation})$. Figure 4.2a shows P_2 in the aggregate graph corresponding to S_2 while figure 4.2b shows its 3 occurrences in the base graph.

4.1.1 Support

As a pattern consists of an edge, a pattern P and a pair of tuples (T_1, T_2) denote the same concept. We mentioned above that we use the actual and the expected number of occurrences of P in the data to further score the interestingness of P. Similarly to the situation of frequent itemset mining, we call the number of occurrences of P the *support* of P and formalize it in definition 4.2.

Definition 4.2. The **support** $supp_S(T_1, T_2)$ of a pattern $P = (T_1, T_2)$ in a cuboid S is defined as the number of occurrences of P. That is the number of edges in the multidimensional network $\mathcal{N} = (V, E, A)$ respecting the conditions of P.

$$supp_S(T_1, T_2) = \sum_{(u,v) \in E} \delta(u, v)$$

with

$$\delta(u, v) = \begin{cases} 1 & \text{if } S(u) = T_1 \wedge S(v) = T_2 \vee S(u) = T_2 \wedge S(v) = T_1 \\ 0 & \text{otherwise} \end{cases}$$

Alternatively it also corresponds to the weight of the edge in the aggregate network $G = (V', E')$ corresponding to the cuboid S.

In example 4.1 the patterns (Female, Lawyer)-(Female, Teacher) and (Lawyer)-(Teacher) occur a single time and 4 times respectively. Therefore their respective supports are 1 and 4.

Then we want to know how to decide whether a pattern is interesting or not. We want a measure taking the expected support of P and the actual support of P as input, and that outputs whether the difference between the two inputs is significant or not. The freedom we have here consists of choosing how to compute the expected support and how to measure the significance.

4.1.2 Expected support

A possible definition for the expected support of a pattern is inspired by the partition models [26] and adapted to the graph cube lattice we are working with. The partition model is a generalization of the independence model that we discussed in section 2.2.1. In an itemset mining setting we assume that all the elements of a partition of the items occur independently from each other. Formally for a given itemset X we can define a partition $\mathcal{P} = P_1, \ldots, P_m$ such that $\bigcup_{i=1}^{m} P_i = X$ and $P_i \cap P_j = \varnothing, i \neq j$. Then the expected frequency of an itemset X is the product of the frequencies of the elements of the partition, i.e. it is equal to

$$\prod_{i=1}^{m} f(P_i)$$

Please note that the independent model is the partition model for which we set one item per partition: $P_i = x_i, i = 1, \ldots, n$. Besides we can choose any partition among the 2^n possible ones for an itemset of size n. Therefore we can compute 2^n expected frequencies for an itemset.

We base our definition of the expected support of a pattern on the idea of a partition model. For a pattern P defined over a cuboid S we express the expected support of P with respect to a more aggregated cuboid $S' \subset S$. Similarly to the itemset mining problem here we can choose $2^{|S|} - 1$ cuboids $S' \subset S$, yielding as many expected supports for P.

First let us 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 always between 0 and 1 and can be seen as the probability $\Pr(T \mid T')$.

Definition 4.3. 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

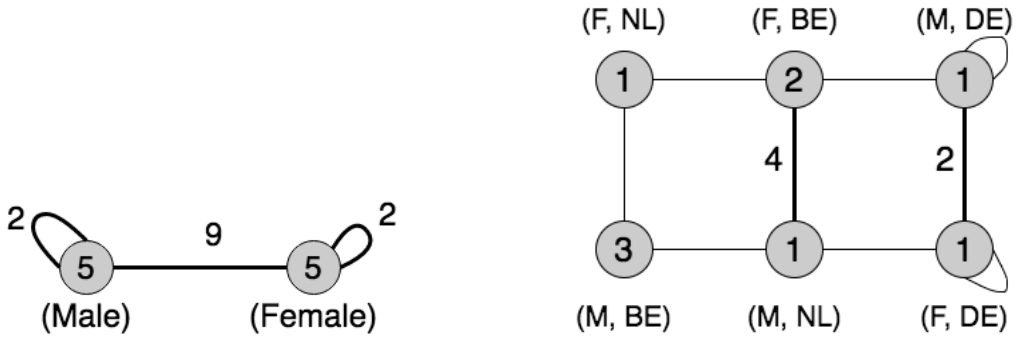
$$p_{S,S'}(T) = \frac{\sum_{u \in V} \gamma_S(u)}{\sum_{u \in V} \gamma_{S'}(u)}$$

with

$$\gamma_S(u) = \begin{cases} 1 & \text{if } S(u) = T \\ 0 & \text{otherwise} \end{cases}$$

Example 4.2. Let us consider the toy network presented in example 1.1. Figures 4.3a and 4.3b present the networks associated with the cuboids $S = (\text{Gender}, \text{Location}, *)$ and $S' = (\text{Gender}, *, *)$ respectively. Let us define W_V and $W_{V'}$ the functions mapping a node to its weight in the networks corresponding to the cuboids S and S' respectively. Then the proportion of the tuple (Female, Belgium) with respect to S' is given by

$$\frac{W_V((\text{F}, \text{BE}))}{W_{V'}((\text{F}))} = \frac{2}{5}$$



(a) (Gender, ∗, ∗) aggregate network (b) (Gender, Location, ∗) aggregate network

Figure 4.3: The (Gender, ∗, ∗) and (Gender, Location, ∗) aggregate networks built from the toy network

The expected support of a pattern $P = (T_1, T_2)$ in a network G corresponding to a cuboid S can be computed from a more aggregated cuboid $S' \subset S$. 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 the fact that we observe P', that is $\Pr(P \mid P')$. We calculate this probability under the null hypothesis that the data is generated from a *partition* 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 endpoints attribute values to match the pattern P given that we know the support of P' is given by equation 4.1

$$\Pr(P \mid P') = p_{S,S'}(T_1)p_{S,S'}(T_2) \tag{4.1}$$

This is the probability that one of the $supp(P')$ edges is an edge between T_1 and T_2. The expected support of P can be seen as $\Pr(P \mid P')$ multiplied by the support of P'.

However this quantity is not always correct. Let us consider the $(\text{Gender}, *, *)$ network in figure 4.3a 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 $\frac{5}{10} \times \frac{5}{10} = 0.25$. 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 endpoints 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.

In general this situation occurs when drawing a non self-loop edge (T_1, T_2) whose matching edge in an ancestor cuboid is a self-loop. Hence we chose to define the probability of observing an undirected edge (T_1, T_2) as 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.

Definition 4.4. The **expected support** $\mathbf{E}[supp_S(T_1, T_2)]$ of a pattern (T_1, T_2) defined over a selection of attributes $S \subseteq A$ is derived from a more aggregated selection of attributes $S' \subset S$. It is given by

$$\mathbf{E}[supp_S(T_1, T_2)] = supp_{S'}(T_1, T_2)p_{S,S'}(T_1)p_{S,S'}(T_2)$$

In example 4.2 we presented two cuboids and their associated networks. Let us consider the pattern $P = ((\mathrm{F}, \mathrm{BE}), (\mathrm{M}, \mathrm{NL}))$ that has a support of 5. Its expected support with respect to $S' = (\mathrm{Gender}, *, *)$ is equal to

$$supp_{S'}((\mathrm{F}, \mathrm{BE}), (\mathrm{M}, \mathrm{NL}))p_{S,S'}((\mathrm{F}, \mathrm{BE}))p_{S,S'}((\mathrm{M}, \mathrm{NL})) = 9 \cdot \frac{2}{5} \cdot \frac{1}{5} = 0.72$$

We define in the next section how to compare the expected and the actual supports of a pattern P to determine whether P is interesting.

4.2 Significance of a pattern

To determine if a pattern P defined over S is interesting or not we can compare the expected support and the actual support of P. As the expected support is computed over a more aggregated view of the network $S' \subset S$, we can choose several cuboids S'. In order not to consider an exponential number of cuboids S' we restrict ourselves to the direct ancestors of S, i.e. the set $S_{anc} = \{S' \subset S \mid |S'| = |S| - 1\}$. For every $S' \in S_{anc}$, we can compute the expected support $\mathbf{E}[supp(P)]_{S'}$. Then we have $|S_{anc}|$ expected supports to compare with the actual support of P. Please note that $|S_{anc}| = |S|$. Sections 4.2.1 and 4.2.2 introduce ad hoc measures to compare actual and expected supports. In section 4.2.3 we propose a statistical approach similar to the one we presented in section 2.2 used in MINI [12].

4.2.1 Absolute difference measure

If no direct ancestor $S' \in S_{anc}$ yields an expected support close enough to the actual support, then we can say that P is interesting. Unfortunately we should define what is close enough. We choose to specify a threshold ϵ on the absolute value of the difference between $supp_S(P)$ and $\mathbf{E}[supp_S(P)]_{S'}$. If one expected support gives a difference that is smaller than ϵ, then P is not interesting because the data *explains* P. It might be the case that the pattern P' itself is interesting. A possible way to define the interestingness of a pattern P defined over S is the following. P is surprising if no expected support defined over $S' \subset S$ can *explain* the actual support of P. That is, if equation 4.2 is satisfied.

$$|\mathbf{E}[supp_S(P)]_{S'} - supp_S(P)| \geq \epsilon \quad \forall S' \in S_{anc} \tag{4.2}$$

This approach suffers from at least one problem. Having the same threshold ϵ for every pattern in any cuboid favors patterns with a high support. For instance if a pattern P_1 with $supp_{S_1}(P_1) = 100$ is best explained by one cuboid S'_1 yielding $\mathbf{E}[supp_{S_1}(P_1)]_{S'_1} = 90$, the minimum difference is equal to 10. Considering another pattern P_2 best explained by S'_2 with $supp_{S_2}(P_2) = 10$ and $\mathbf{E}[supp_{S_2}(P_2)]_{S'_2} = 9$, its minimum difference is equal to 1. Although both P_1 and P_2 supports differ from their respective best explaining expected support by 10%, P_1 is reported as more interesting than P_2.

4.2.2 Ratio measure

To avoid the situation depicted in section 4.2.1, we can normalize the differences present in equation 4.2. They have the form $|x - y|$. Dividing a difference by $\max(x, y)$ results in a quantity between 0 and 1. Therefore in our situation described above, P_1 and P_2 become equally

Pattern	Support	Expected support(s)	Difference(s)	Ratio(s)
(M)-(F)	9	6.5	2.5	0.28
(M, Eng)-(M, Law)	2	0.96, 1.33	1.04, 0.67	0.52, 0.33

Table 4.1: Differences and ratios for two patterns in the toy network

interesting using this ratio measure. Equation 4.3 gives the updated quantity to measure for a pattern P, i.e. the ratio of P. Then definition 4.5 presents an updated formalization of the interestingness of P.

$$r_{S,S'}(P) = \frac{|supp_S(P) - \mathbf{E}[supp_S(P)]_{S'}|}{\max(supp_S(P), \mathbf{E}[supp_S(P)]_{S'})} \tag{4.3}$$

Definition 4.5. A pattern P defined over S is **interesting** if no expected support defined over $S' \subset S$ can *explain* the actual support of P. That is, if

$$r_{S,S'}(P) \geq \rho \quad \forall S' \in S_{anc}$$

Fixing a threshold ρ between 0 and 1 allows us to search for patterns whose support deviate by at least ρ% from their closest expected support. These patterns are labeled as interesting under our model, and can be ordered by how surprising they are. The measure of interestingness of a pattern P denoted by $int(P)$ is given by the minimum ratio quantity $r_{S,S'}(P)$ stated in definition 4.5 among the different ancestors. It corresponds to the least deviation in support with respect to the support P would have under the null hypothesis. Formally $int(P)$ is given in equation 4.4.

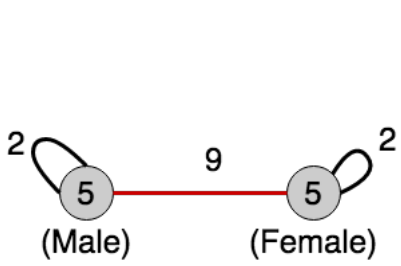
$$int(P) = \min_{S' \in S_{anc}} r_{S,S'}(P) \tag{4.4}$$

Example 4.3. Considering the network in example 1.1 we observe that the edge (Male)-(Female) denoted by P_1 in the cuboid $S_1 = (\text{Gender}, *, *)$ shown in figure 4.3a has a support of 9. As S_1 has a single ancestor being the apex, it has a single expected support. Let us consider a second aggregate edge, (Male, Engineer)-(Male, Lawyer) denoted by P_2 represented in the cuboid $S_2 = (\text{Gender}, *, \text{Profession})$ in figure 4.3b. It has a support of 2 and two expected weights. The absolute differences and the ratios of the two edges are computed respectively by equations 4.2 and 4.3. These values are presented in table 4.1. Please note that the best explaining expected support is 1.33 yielding a difference of 0.67 and a ratio of 0.33.

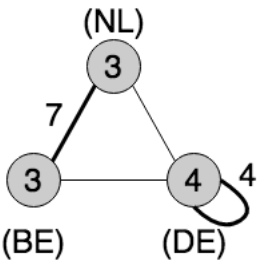
We observe that even though P_1 has an absolute difference quite higher than the one of P_2, their ratios are almost equal. This illustrates the need of the ratio $r_{S,S'}(P)$ measure.

4.2.3 Statistical approach

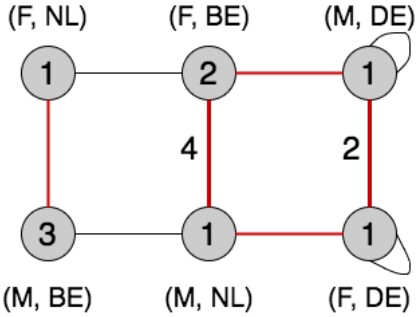
Although the interestingness of a pattern as we defined it in section 4.2.2 can be used to rank patterns, this is a simple comparison of two quantities, the expected and the actual supports. A statistical approach is more well founded and brings a point of view based on known axioms. We will derive the probability that a pattern P has indeed a support equal to the observed support $supp(P)$ given that P is drawn from its *ancestor edge* in an ancestor. Indeed if we consider an edge P in the network G corresponding to a cuboid S and a certain ancestor $S' \subset S$, P has a single matching edge in G' corresponding to S'. 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.



(a) (Gender, ∗, ∗) aggregate network



(b) (∗, Location, ∗) aggregate network



(c) (Gender, Location, ∗) aggregate network

Figure 4.4: The networks corresponding to the cuboids (Gender, ∗, ∗), (∗, Location, ∗) and (Gender, Location, ∗) built from the toy network

Let p be the probability that one of the edges P' in the ancestor S' is an edge matching P as computed in equation 4.1. Since the edges P' are distributed in S, we can model the probability of observing P given P' by a multinomial distribution. For example considering $P' = ((\text{Male}), (\text{Female}))$ highlighted in figures 4.4a and 4.4c, 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 $P_1 = ((\text{Female, BE}), (\text{Male, NL}))$ is drawn from the 9 edges is given by $\frac{2}{5} \cdot \frac{1}{5} = 0.08$. Please note that $supp(P_1) = 4$ and that P_1 is as likely to be drawn as $P_2 = ((\text{Female, BE}), (\text{Male, DE}))$ while $supp(P_2) = 1$.

We see an edge in S as a random variable with a probability of success or probability to be present p with respect to an ancestor S'. Then the probability to observe the actual support can be modeled by the means of the multinomial distribution defined over all the possible edges in S that can be matched to a particular edge in S'. Equation 4.5 defines the probability to observe $supp(P)$ in a cuboid S with respect to an ancestor cuboid $S' \subset S$. The multidimensional network corresponding to S' is given by $G' = (V', E', A')$.

$$\Pr(supp(P)) = \binom{|E'|}{supp(P)} p^{supp(P)} (1-p)^{|E'|-supp(P)} \tag{4.5}$$

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 aggregated 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'.

Cuboid	Pattern	Absolute difference
(Location)	(BE)-(NL)	4.66
(Occupation)	(Eng)-(Tea)	2.88
(Gender)	(M)-(F)	2.50
(Location)	(NL)-(DE)	2.12
(Location)	(BE)-(DE)	2.12
(Location)	(DE)-(DE)	1.92
(Gender)	(F)-(F)	1.25
(Gender)	(M)-(M)	1.25
(Occupation)	(Tea)-(Tea)	1.08

Table 4.2: Top 10 most interesting patterns in the toy network using the absolute difference measure

Example 4.4. Let us consider the pattern $P = ((\text{Female, BE}), (\text{Male, NL}))$ whose context is given in figure 4.4. The probability to effectively observe 4 such edges given the (Gender, $*$, $*$) network given in figure 4.4a is equal to

$$\binom{9}{4} 0.08^4 (0.92)^5 = 0.003.$$

We can also compute the probability to observe these 4 edges given the ($*$, Location, $*$) network given in figure 4.4b. This probability is equal to

$$\binom{7}{4} \left(\frac{2}{3}\right)^4 \left(\frac{1}{3}\right)^3 = 0.25$$

We can adapt the interestingness measure $int(P)$ of equation 4.4 to our statistical approach. Instead of searching for the minimum deviation ratio between the actual and the expected supports we are looking for the highest probability to observe the actual support under the independence model. In both cases we define $int(P)$ as the quantity that best *explains* the observed support of P according to the independence model. Equation 4.6 defines the interestingness of a pattern P using our statistical approach.

$$int(P) = \max_{S' \in S_{anc}} \Pr(supp(P)) \tag{4.6}$$

We calculated the probability of observing $supp(P)$ for $P = ((\text{Female, BE}), (\text{Male, NL}))$ given each ancestor network in example 4.4. This pattern is best explained by the ($*$, Location, $*$) network with a probability of 0.25. Hence the interestingness of P is equal to 0.25.

4.3 Significance measures comparison

In this section we compare the three approaches we defined in section 4.2 to rank the most interesting patterns in the toy network. We report in tables 4.2, 4.3 and 4.4 the top 10 most interesting patterns using the absolute difference measure, the ratio measure and the statistical approach respectively.

As we mentioned in chapter 3, the patterns of interest are the high number of connections between Belgians and Dutchmen and the high number of connections between pairs of Germans.

The absolute difference measure favors patterns with a much higher or a much lower support than expected under the independence model. We can see this bias as all the reported patterns

Cuboid	Pattern	Ratio
(Location)	(NL)-(DE)	0.68
(Location)	(BE)-(DE)	0.68
(Location)	(BE)-(NL)	0.67
(Occupation)	(Tea)-(Tea)	0.52
(Gender, Location, Occupation)	(M, DE, Law)-(F, DE, Law)	0.50
(Gender, Location, Occupation)	(F, DE, Tea)-(F, DE, Law)	0.50
(Gender, Location, Occupation)	(M, DE, Eng)-(M, DE, Law)	0.50
(Location, Occupation)	(DE, Eng)-(DE, Tea)	0.50
(Location, Occupation)	(NL, Eng)-(DE, Tea)	0.50
(Occupation)	(Eng)-(Tea)	0.48

Table 4.3: Top 10 most interesting patterns in the toy network using the ratio measure

Cuboid	Pattern	Probability
(Location)	(BE)-(NL)	3.19×10^{-3}
(Occupation)	(Eng)-(Tea)	0.05
(Gender)	(M)-(F)	0.09
(Location)	(DE)-(DE)	0.10
(Location)	(NL)-(DE)	0.12
(Location)	(BE)-(DE)	0.12
(Gender, Occupation)	(M, Eng)-(F, Tea)	0.14
(Gender)	(F)-(F)	0.21
(Gender)	(M)-(M)	0.21
(Gender, Location)	(M, NL)-(F, BE)	0.23

Table 4.4: Top 10 most interesting patterns in the toy network using the statistical approach

belong to a network defined over a single attribute. These networks contains highest number of edges in the lattice. The (BE)-(NL) is reported as the best pattern because its support high and moreover because it is much higher than expected. The relationships (NL)-(DE) and (BE)-(DE) being the only two edges between a German and a person from another country, they are also reported. However the highly surprising connection (DE)-(DE) is at the sixth place with a measure less than the half of the best pattern.

The ratio measure reports three surprising (Location) patterns but fails to report the (DE)-(DE) connection. It is due to some individual edges having a low expected support. The ratio measure tends to report patterns in aggregate networks defined over more attributes.

The statistical approach reports (BE)-(NL) as the most surprising pattern. The (Engineer)-(Teacher) relationships are ranked second. It is indeed surprising to observe 6 connections of this kind from the 13 edges as they could have been better distributed. The association (DE)-(DE) is at the fourth place with a measure close to the second and third best patterns. The pattern (M, Engineer)-(F, Teacher) is also marked as surprising. Indeed in the toy network all the engineers are men and all the teachers are women. This situation can be interpreted as surprising.

In summary, the absolute difference measure favors patterns in the more aggregated networks while the ratio measure is the only one to report patterns in the base network. The statistical approach reports high level patterns as the absolute difference measure but also patterns in lesser aggregated networks when it is relevant.

In chapter 3 we discussed the top 10 most informative and non-redundant patterns reported by MINI [12]. These patterns are presented in table 3.2b. No pattern defined over a single dimension is reported. However the 2 best patterns are (M, Engineer)-(F, Teacher), (F, Teacher)-(F, Teacher). As we mentioned above the first connection is indeed surprising. The second one involves 4 women that are teachers and that formed only one aggregate edge among them. But MINI should report itemsets that have a higher support than expected. In this case it is the opposite, we expect a higher number of connections. This situation illustrates one limitation of our mapping: a self-loop in any of the aggregate networks is present two times because each transaction is translated into two edges. MINI also reports the pattern (DE, Lawyer)-(DE, Lawyer). But it would be more interesting to report the connection (DE)-(DE) our toy network contains a cluster of German people having connections among theme.

4.4 Search algorithm

In this section we present in details how we perform the search for interesting aggregate edges. These edges are found in any of the networks contained in the graph cube lattice $\mathcal{L}$ built from a given multidimensional network $\mathcal{N}$.

Algorithm 1 presents our search procedure which consists of an exhaustive search throughout every aggregate network. It takes as input the lattice $\mathcal{L}$ and a probability threshold θ. This threshold is used to determine whether an aggregate edge is interesting or not. Indeed if the interestingness of a pattern P in the sense of equation 4.6 is greater than θ then P is not interesting. We say that one of the direct ancestors of the network in which P lies *explains* P with respect to θ.

Each aggregate edge of each aggregate network is considered once as a candidate of interest. Let us denote by P a candidate edge of a network $\mathcal{N}$ considered during the search process. If one of the direct ancestors of $\mathcal{N}$ *explains* the support of P with respect to θ then P is not added to the set of interesting patterns. On the other hand if no direct ancestor can explain P then it is marked as interesting.

4.4.1 Complexity analysis

Our miner performs an exhaustive search in the lattice $\mathcal{L}$. As $\mathcal{L}$ contains an exponential number of cuboids the time complexity is also exponential. Let $\mathcal{N} = (V, E, A)$ be the given base multidimensional network. Let $n = |A|$ be the size of the dimension of $\mathcal{N}$ and let $m = |E|$ be the number of edges in $mathcalN$. The number of cuboids in $\mathcal{L}$ is 2^n. The search process goes through each edge of each aggregate network in $\mathcal{L}$. Furthermore for each edge it checks at most n direct ancestors. Indeed each cuboid having a dimension of size x has x ancestors because each attribute can be aggregated. Since it is the base graph that has the highest size in dimension we take n as the upper bound on the number of direct ancestors to consider per edge. At the end of the day the time complexity of algorithm 1 is $\mathrm{O}(mn2^n)$.

In practice the polynomial time part of this time complexity is often reduced because the number of direct ancestors decreases as the considered cuboid is close to the apex in the lattice. This is also the case for the number of edges in the aggregate networks. The closer the corresponding cuboid is to the apex the smaller is the number of edges. The worst case scenario is a network whose nodes cannot be merged in any of the aggregate networks. This means that for every attribute, every node has a different attribute value from the other nodes. As an example let us consider a network representing the relationships between people given their age, location and profession. In the worst case scenario every person has a different age, a different location and a different profession. There is no pair of nodes such that they can be

aggregated. However, in such a scenario, there is nothing to report as interesting since everyone is independent from the others.

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

```
1  patterns ← ∅
2  for 𝒩 ← ℒ \ apex do
3      (V, E, A) ← 𝒩
4      for e ∈ E do
5          interesting = true
6          ancestors = {(V', E', A') | A' ∈ A, |A'| = |A| − 1}
7          for 𝒩' ← ancestors do
8              p = probability(𝒩, 𝒩', e)
9              if p ≥ θ then
10                 interesting = false
11                 break
12             end
13         end
14         if interesting then
15             patterns ← patterns ∪ (e, p)
16         end
17     end
18 end
19 return patterns sorted by probability
```

Chapter 5

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. Such a user can rate any movie available in the website database in order to get refined movie recommendations. We use it to build a graph where the nodes correspond to users and we add 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 chapter how we construct these two attributed graphs and the experiments we conducted on them.

5.1 Graph construction

5.1.1 The MovieLens dataset

The GroupLens research group proposes several datasets they produced over the existence of the MovieLens website [16]. We choose to use the MovieLens1M dataset because it encompasses some demographic information about the users, namely their age, gender, occupation and location. The dataset consists of a million of ratings given by one of the 6,000 users to one of the 3,700 movies between 2000 and 2003. The rating scale is an integer from 1 to 5 stars. To be included each user must have given at least 20 ratings in addition to her personal information. As users that did not rate enough movies are not included in the dataset, one limitation is a bias towards "prolific" users, i.e. users that were able to find enough content to rate.

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 (`age`, `gender`, `occupation`, `location`). 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 build.

[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`

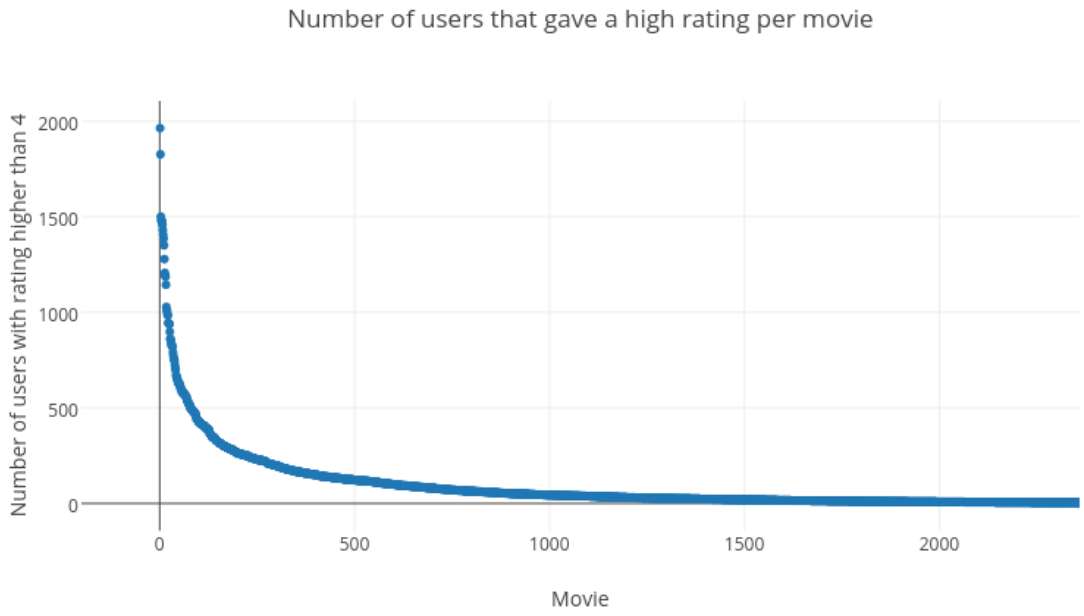


Figure 5.1: Number of high ratings per movie in the MovieLens1M dataset

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 with a 4 or a 5, considering that a 3 is not sufficient. 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 5.1 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 5.1 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 5.2 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 final network contains 1393 movies distributed in 975 nodes and 32537 similarities higher than 20 distributed in 27150 edges.

5.1.2 The extended MovieLens dataset

The second International Workshop on Information Heterogeneity and Fusion in Recommender Systems [7] (HetRec 2011) released several datasets among which the extended MovieLens dataset.

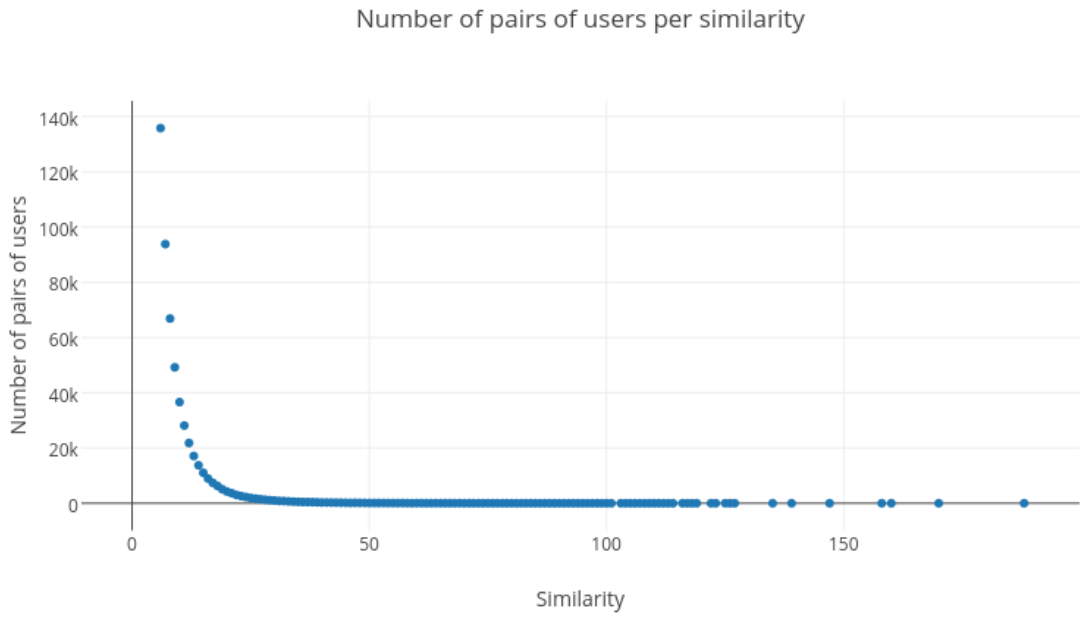


Figure 5.2: Number of pairs of users per similarity in the pruned MovieLens1M dataset

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. This dataset contains 850,000 ratings from 2100 users on 10,200 movies. 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. We also expect the function depicting the number of movies rated 4 or more per user to follow a power law. The empirical results presented in figure 5.3 confirm this fact since we can observe that a few users took the time to rate a high number of movies or like a lot of them while 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, as we observe an abrupt slope around 400 movies in figure 5.3.

Then the similarity threshold is empirically defined in a similar manner as we did for the MovieLens1M dataset. Figure 5.4 presents the number of pairs of movies per similarity. We removed similarities 1 to 5 to improve readability. We can observe an abrupt slope around a similarity of 40. Once again in order to decrease the number of edges in the graph we set the similarity threshold to 40. The final network contains 678 movies distributed in 318 nodes and 25808 similarities higher than 40 distributed in 10985 edges.

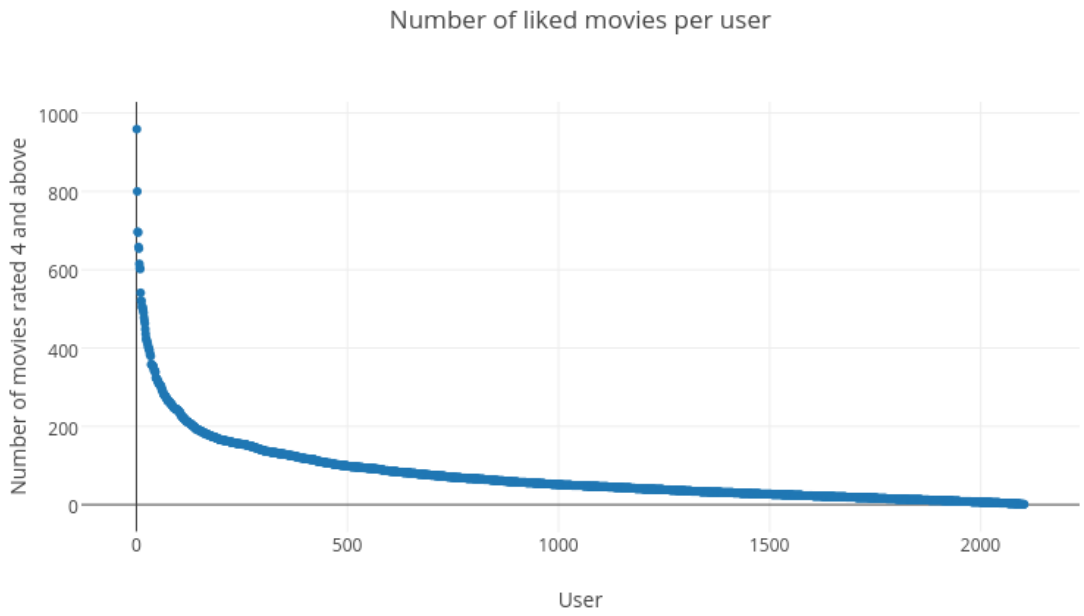


Figure 5.3: Number of movies rated 4 or more per user in the extended MovieLens dataset

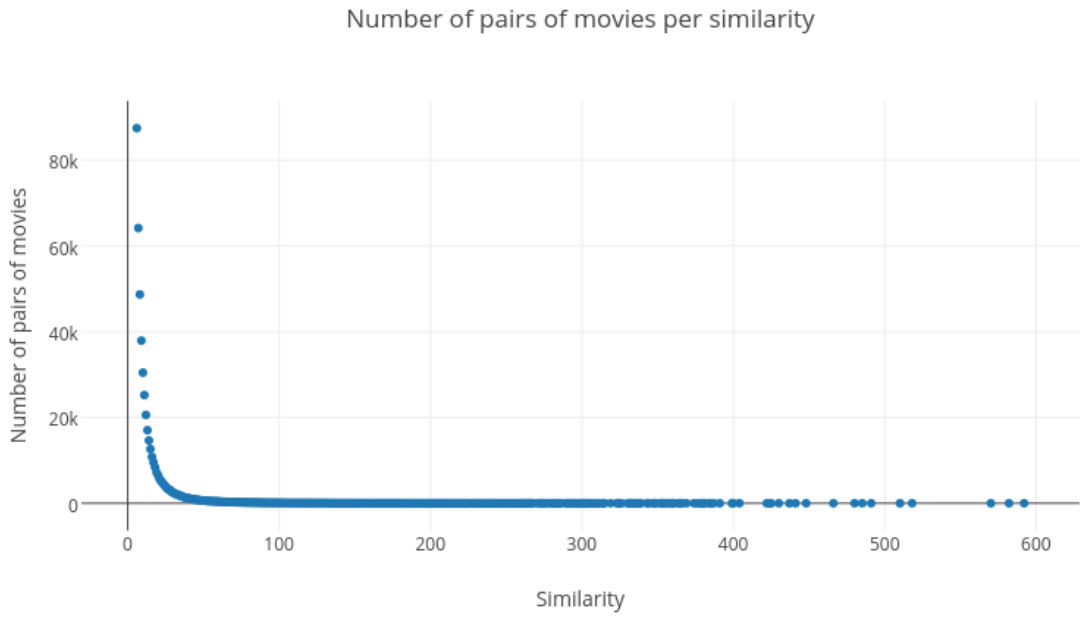


Figure 5.4: Number of pairs of movies per similarity in the extended MovieLens dataset

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 5.1: Top 10 most interesting edges in the MovieLens1M dataset reported by our miner

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

5.2 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 5.1. 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 5.2. 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 tables 5.3 and 5.4. 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**	**p-value**
(Age)	(25)-(25)	0.74
(Gender)	(M)-(F)	0.74
(Gender)	(F)-(M)	0.74
(Gender)	(M)-(M)	0.74

Table 5.3: 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 5.4: Top 10 most interesting patterns in the extended Movielens10M dataset reported by MINI

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 algorithms MINI and MTV 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.

Bibliography

[1] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *Acm sigmod record*, volume 22, pages 207–216. ACM, 1993.

[2] Rakesh Agrawal, Heikki Mannila, Ramakrishnan Srikant, Hannu Toivonen, A Inkeri Verkamo, et al. Fast discovery of association rules. *Advances in knowledge discovery and data mining*, 12(1):307–328, 1996.

[3] Leman Akoglu, Mary McGlohon, and Christos Faloutsos. Oddball: Spotting anomalies in weighted graphs. *Advances in Knowledge Discovery and Data Mining*, pages 410–421, 2010.

[4] Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based anomaly detection and description: a survey. *Data Mining and Knowledge Discovery*, 29(3):626–688, 2015.

[5] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[6] Douglas Burdick, Manuel Calimlim, and Johannes Gehrke. Mafia: A maximal frequent itemset algorithm for transactional databases. In *Data Engineering, 2001. Proceedings. 17th International Conference on*, pages 443–452. IEEE, 2001.

[7] Iván Cantador, Peter Brusilovsky, and Tsvi Kuflik. 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *Proceedings of the 5th ACM conference on Recommender systems*, RecSys 2011, New York, NY, USA, 2011. ACM.

[8] Deepayan Chakrabarti. Autopart: Parameter-free graph partitioning and outlier detection. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 112–124. Springer, 2004.

[9] Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 1991.

[10] Tijl De Bie. Maximum entropy models and subjective interestingness: an application to tiles in binary databases. *Data Mining and Knowledge Discovery*, 23(3):407–446, 2011.

[11] William Eberle and Lawrence Holder. Anomaly detection in data represented as graphs. *Intelligent Data Analysis*, 11(6):663–689, 2007.

[12] Arianna Gallo, Tijl De Bie, and Nello Cristianini. Mini: Mining informative non-redundant itemsets. In *European Conference on Principles of Data Mining and Knowledge Discovery*, pages 438–445. Springer, 2007.

[13] Jing Gao, Feng Liang, Wei Fan, Chi Wang, Yizhou Sun, and Jiawei 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*, pages 813–822. ACM, 2010.

[14] Floris Geerts, Bart Goethals, and Taneli Mielikäinen. Tiling databases. In *International Conference on Discovery Science*, pages 278–289. Springer, 2004.

[15] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[16] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems (TiiS)*, 5(4):19, 2016.

[17] Edwin T Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.

[18] Kleanthis-Nikolaos Kontonasios and Tijl De Bie. An information-theoretic approach to finding informative noisy tiles in binary databases. In *Proceedings of the 2010 SIAM international conference on data mining*, pages 153–164. SIAM, 2010.

[19] Michael Mampaey, Nikolaj Tatti, and Jilles 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*, pages 573–581. ACM, 2011.

[20] Caleb C Noble and Diane J Cook. Graph-based anomaly detection. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 631–636. ACM, 2003.

[21] Nicolas Pasquier, Yves Bastide, Rafik Taouil, and Lotfi Lakhal. Discovering frequent closed itemsets for association rules. In *International Conference on Database Theory*, pages 398–416. Springer, 1999.

[22] Gideon Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464, 1978.

[23] Claude Elwood Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55, 2001.

[24] Craig Silverstein, Sergey Brin, and Rajeev Motwani. Beyond market baskets: Generalizing association rules to dependence rules. *Data mining and knowledge discovery*, 2(1):39–68, 1998.

[25] Takeaki Uno, Tatsuya Asai, Yuzo Uchida, and Hiroki Arimura. Lcm: An efficient algorithm for enumerating frequent closed item sets. In *FIMI*, volume 90. Citeseer, 2003.

[26] Jilles Vreeken and Nikolaj Tatti. Interesting patterns. In *Frequent pattern mining*, pages 105–134. Springer, 2014.

[27] Mohammed J Zaki and Ching-Jui Hsiao. Charm: An efficient algorithm for closed itemset mining. In *Proceedings of the 2002 SIAM international conference on data mining*, pages 457–473. SIAM, 2002.

[28] Peixiang Zhao, Xiaolei Li, Dong Xin, and Jiawei Han. Graph cube: on warehousing and olap multidimensional networks. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pages 853–864. ACM, 2011.

Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve **www.uclouvain.be/epl**