EXTRACÇÃO DE RELAÇÕES SEMÂNTICAS EM INFORMAÇÃO NÃO ESTRUTURADA PARA ENRIQUECIMENTO DE ONTOLOGIAS DE DOMÍNIO

SEMANTIC RELATIONS EXTRACTION IN UNSTRUCTURED INFORMATION FOR DOMAIN ONTOLOGIES ENRICHMENT

Abstract

Since semantic web appearance, several domain ontologies were developed and delivered in open access repositories. The existing domain ontologies describe semantic elements specific to a particular domain. These elements can be used complementing existing document information. This articulation can be duly empowered if, new methods could be created that in a semi automatic way, could help the ontologic precision. Specifically, the new patterns originating the building of new knowledge will be extracted not only from domain ontologies, but also from unstructured information sources.

One of the greatest challenges related to domain ontology enrichment, known in scientific community as “ontology learning”, is the fact in which “pure” automated processes that could make the above mentioned enrichment from unstructured information sources does not exist. In scientific community there are several contributions in this area, namely in the development of methods to quantify the way that existent ontology concepts inside domain ontology are related. These approaches only take advantage of the information included in ontologies and do not consider exterior information to quantify these relations.

The main goal of this dissertation is to use data mining techniques as a way of extracting patterns (here presented as semantic associations) from unstructured information sources. The idea to develop in this work is based on the statistic analysis of co-occurrence between, the more relevant terms from a document corpus and to quantify this analysis through semantic relations between concepts from domain ontology. The domain of the information sources presented in this project is focused on Civil Construction.

Keywords:

Sumário

Desde o aparecimento da *web* semântica, várias ontologias de domínio foram desenvolvidas e disponibilizadas em repositórios de acesso aberto. As ontologias de domínio existentes descrevem elementos semânticos, que são específicos a um determinado domínio, elementos esses que podem ser usados com forma de complemento à informação existente em documentos. Esta complementaridade poderá ser devidamente potenciada, se forem criados novos métodos que de forma semi-automática auxiliarem o refinamento ontológico. Mais especificamente, os novos padrões que dão origem à geração novo conhecimento poderão ser extraídos não só de ontologias de domínio, mas também de fontes de informação não estruturada.

Um dos grandes desafios relacionados com o enriquecimento de ontologias de domínio designado na comunidade científica por *ontology learning*, prende-se com o facto não existirem automatismos “puros” que permitam esse mesmo enriquecimento a partir de fontes de informação não estruturadas. Existem bastantes contribuições científicas nesta área, nomeadamente no desenvolvimento de métodos que permitam quantificar a forma como os conceitos existentes numa ontologia de domínio estão relacionados. Estas abordagens utilizam apenas a informação contida nas ontologias e não fazem uso de informação externa à ontologia para quantificar essas mesmas relações.

Esta dissertação tem como principal objectivo, o uso de técnicas de *data mining* como forma de extracção de padrões (aqui definidos para associações semânticas) em fontes de informação não estruturada. A ideia a ser desenvolvida no âmbito desta dissertação, tem por base a análise estatística da co-ocorrência entre os termos mais relevantes de um corpus de documentos e, quantificar essa análise sob a forma de relações semânticas entre conceitos de uma ontologia de domínio. O domínio das fontes de informação aqui a serem tratadas, é focado no sector da construção civil.

Palavras-Chave:

*Dedico a concretização desta etapa, finalizada por esta dissertação aos meus Pais, Mário Luiz e Maria Edite…*

*“Always look on the bright side of life!”*

*Monty Python, in “Life of Brian”*

Contents

[1 Introduction 2](#_Toc396690220)

[1.1 Motivation 3](#_Toc396690221)

[1.2 Vision 3](#_Toc396690222)

[1.3 Goals 4](#_Toc396690223)

[1.4 Development context 4](#_Toc396690224)

[1.5 Dissertation Structure 4](#_Toc396690225)

[2 State of the Art / Related Work 6](#_Toc396690226)

[2.1 Concept Relation Quantification 6](#_Toc396690227)

[2.2 Ontology Learning 6](#_Toc396690228)

[2.2.1 Association rules 6](#_Toc396690229)

[2.3 Building & Construction 6](#_Toc396690230)

[3 Theoretical and Technical Foundation 8](#_Toc396690231)

[3.1 Association Rules 8](#_Toc396690232)

[3.1.1 Frequent Pattern Growth 11](#_Toc396690233)

[3.1.2 Association Rules Measurement 19](#_Toc396690234)

[3.2 Vector Space Model 25](#_Toc396690235)

[3.2.1 Term Weighting – The TF-IDF 26](#_Toc396690236)

[3.3 Similarity Measure in Information Retrieval 27](#_Toc396690237)

[3.3.1 Cosine Similarity Measure Algorithm 28](#_Toc396690238)

[4 Building & Construction Domain Ontology 30](#_Toc396690239)

[4.1 Ontology 30](#_Toc396690240)

[4.1.1 Construction Methodology 31](#_Toc396690241)

[4.1.2 Ontologic enrichment Dynamics 31](#_Toc396690242)

[4.1.3 The E-COGNOS Project – Ontology in Building and Construction 31](#_Toc396690243)

[5 Design and Implementation 32](#_Toc396690244)

[5.1 Tools and Technologies 32](#_Toc396690245)

[5.2 Requirements 33](#_Toc396690246)

[5.3 Conceptual & Technical Architectures 34](#_Toc396690247)

[5.4 Implementation and Development 35](#_Toc396690248)

[5.5 Conceptual & Technical Architectures 36](#_Toc396690249)

[5.5.1 Document Analysis 37](#_Toc396690250)

[5.5.2 FP-Growth 39](#_Toc396690251)

[5.5.3 Association Rules 39](#_Toc396690252)

[5.5.4 Frequent Itemset Mapping 39](#_Toc396690253)

[5.5.5 Ontology Enrichment 40](#_Toc396690254)

[5.6 Front end 40](#_Toc396690255)

[6 Evaluation 42](#_Toc396690256)

[6.1 Use cases 42](#_Toc396690257)

[6.2 Scientific publications 42](#_Toc396690258)

[7 Conclusion and Future Work 44](#_Toc396690259)

[8 Bibliography 46](#_Toc396690260)

[9 Appendices 50](#_Toc396690261)

[Figure 3.1 – An FP-Tree example for the items in the transaction Table 3‑1. 14](file:///F:\Os%20meus%20documentos\Universidade\Dissertação\Escrita\working\inprocess\Dissertação%20v4%20(Tudo).docx#_Toc396690262)

[Figure 3.2 – The three initial trees at the end of the first three transactions 15](file:///F:\Os%20meus%20documentos\Universidade\Dissertação\Escrita\working\inprocess\Dissertação%20v4%20(Tudo).docx#_Toc396690263)

[Figure 3.3 – Prefix sub-paths for all frequent items 16](file:///F:\Os%20meus%20documentos\Universidade\Dissertação\Escrita\working\inprocess\Dissertação%20v4%20(Tudo).docx#_Toc396690264)

[Figure 3.4 – Conditional FP-Tree for item *professor* 18](file:///F:\Os%20meus%20documentos\Universidade\Dissertação\Escrita\working\inprocess\Dissertação%20v4%20(Tudo).docx#_Toc396690265)

[Figure 3.5 – Interestingness measures types tree (adapted from (Silberschatz and Tuzhilin, 1995)) 20](#_Toc396690266)

[Figure 5.1 - Tools and Technologies adopted 32](file:///F:\Os%20meus%20documentos\Universidade\Dissertação\Escrita\working\inprocess\Dissertação%20v4%20(Tudo).docx#_Toc396690267)

[Figure 5.2 - System requirements 34](#_Toc396690268)

[Figure 5.3 – Knowledge layer architecture 34](#_Toc396690269)

[Figure 5.4 – System Architecture – MVC Methodology 35](#_Toc396690270)

[Figure 5.5 – Rapidminer Main Process 36](#_Toc396690271)

[Figure 5.6 – Conceptual Architecture 36](#_Toc396690272)

[Figure 5.7 – Vector Creation Rapidminer Process 37](#_Toc396690273)

[Figure 5.8 – Document Analysis Pipeline 38](#_Toc396690274)

[Figure 5.9 – Entity Relation Model 40](#_Toc396690275)

[Table 3‑1 – Transaction table for frequent items in database 12](#_Toc396690276)

[Table 3‑2 – Paths table for frequent items 16](#_Toc396690277)

[Table 3‑3 – Frequent itemsets discovered for all items 18](#_Toc396690278)

[Table 5‑1 – Numerical to Binomial regulation 38](#_Toc396690279)

[Table 5‑2 – Association Rules Database Structure 39](#_Toc396690280)

**SYMBOLS AND NOTATION**

|  |  |  |
| --- | --- | --- |
|  | **AI** | **A**rtificial **I**ntelligence |
|  | **API** | **A**pplication **P**rogramming **I**nterface |
|  | **AR** | **A**ssociation **R**ules |
|  | **ASP** | **A**ctive **S**erver **P**ages |
|  | **CSS** | **C**ascading **S**tyle **S**heet |
|  | **ECLAT** | **E**quivalent **CLA**ss **T**ranformation |
|  | **ERD** | **E**ntity **R**elation **D**iagram |
|  | **FI** | **F**requent **I**tem |
|  | **FP** | **F**requent **P**attern |
|  | **HTML** | **H**yper **T**ext **M**arkup **L**anguage |
|  | **IR** | **I**nformation **R**etrieval |
|  | **IT** | **I**nformation **T**echnology |
|  | **JDBC** | **J**ava **D**ata**B**ase **C**onnection |
|  | **JSP** | **J**ava **S**erver **P**age |
|  | **KD** | **K**nowledge **D**iscovery |
|  | **OWL** | **W**eb **O**ntology **L**anguage |
|  | **RDF** | **R**esource **D**escription **F**ramework |
|  | **SQL** | **S**tructured **Q**uery **L**anguage |
|  | **TF-IDF** | **T**erm **F**requency – **I**nverse **D**ocument **F**requency |
|  | **TID** | **T**ransaction **ID**entification |
|  | **UML** | **U**nified **M**odelling **L**anguage |
|  | **USD** | **U**ML **S**tate **D**iagram |
|  | **USQD** | **U**ML **S**e**Q**uence **D**iagram |
|  | **UUC** | **U**ML **U**se **C**ase |
|  | **VSM** | **V**ector **S**pace **M**odel |
|  | **W3C** | **W**orld **W**ide **W**eb **C**onsortium |
|  | **XML** | e**X**tensible **M**arkup **L**anguage |
|  | **JDOM** | **J**ava **D**ocument **O**bject **M**odel |
|  | **JVM** | **J**ava **V**irtual **M**achine |
|  | **MVC** | **M**odel **V**iew **C**ontroller |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Introduction

In the modern world nowadays, information value is huge. The amount available to everyone is such as the ease that it reaches to every person. The global and exponential growth use of computers by people and businesses, and the rise of the internet changed the way to look to this information and the forms to collect, select and distribute. Every day, in most human daily tasks related to a computer, information records are created. It is thus more important to arrange techniques to represent such information in the form of knowledge representation understandable for the machine and human at the same time. The discovery of these techniques presents a big challenge to engineers. They have to know that the knowledge must be in a way that is searchable, with quick access to minimize the time of the access, making this task almost transparent to users. If there is a way to discover the knowledge desired, objectively and efficient, more time would be available to other tasks as or more important to everyday life.

Today, is almost impossible to live without information and their several representations, if it is a simple daily task or a simple supermarket visit, or even a more complex situations like constructing a building, the information is always present. In any situation also time is very important, so it becomes fundamental for engineers to create means to reduce the access time to knowledge. Through models that respond to this needs, like text and data mining models.

But to achieve an objective result of a search, the knowledge must have some kind of organization. It is not enough to have the knowledge as it is received as raw material. It will be necessary to process it in any way. Like separate it by themes or measure its similarity to a central subject, but the important is to make some processing to help achieve the right results faster.

Através da extracção de conhecimento, a matéria-prima passa por três processos: extracção de palavras, filtragem de expressões regulares e criação do vector estatístico.

A criação de uma ontologia aparece da necessidade de detecção, extracção e relacionamento dos conceitos das diferentes áreas, através de uma forma de classificação. Estas ontologias normalmente são criadas e mantidas de uma forma pela acção de um humano. Reconhecer um conceito, saber se já existe na ontologia, definir um relacionamento, atribuir uma classificação são as tarefas de um responsável pela manutenção de uma ontologia. Isto apresenta-se como um trabalho muito exaustivo, quanto maior for a base de conhecimento ou a ontologia. Algumas questões surgem de imediato. Como reconhecer um conceito, como verificar se ele existe na base de dados, como saber qual a relação que o conceitos tem com outros conceitos, como saber se se relaciona mais com um conceito A do que com um conceito B, como definir uma métrica, como saber se a relação existente é a adequada. As dúvidas que surgem são muitas na cabeça do engenheiro de ontologias?

Uma das motivações deste estudo é a criação de algum dinamismo uma ontologia, no meu entendimento, aparece

Esta é uma das motivações deste trabalho. Ajudar a que uma pesquisa seja o mais objectiva possível para que se possa descobrir rapidamente o conhecimento que deseja e deixar mais tempo para o que é mesmo importante.

Torna-se assim importante falar nas técnicas de web mining, mais concretamente Text mining, e knowlege discovery.

Antes de saber como se descobrir novo conhecimento é necessário saber como é que os sistemas de informação reconhecem o conhecimento quando este está na forma de texto. Existem algumas formas de representação do conhecimento, uma delas chama-se ontologia.

## Motivation

Information is everywhere. Nowadays, every area have a database or repository with information. As the IT systems grow, and the time passes the information also grows, and the complexity of the information sometimes reach sizes that humans do not imagine, neither can deal with them. Besides knowing the human brain is a “machine” that can store lots of knowledge inside, there is no one that have all the information in the world.

With the appearance of Internet and computers, arise the opportunity to store knowledge and share it with others, making the human more aware of the world around. One can be, for instance in Australia, and get an information of Portugal without travelling to the country.

Storing the information makes new challenges for the engineers. With the help of the improving of technology, and the massification of knowledge, the issue of storing information get to a point where was necessary to organize it.

In the competitive engineering world, a good organized system could be a key to reach success. The need of getting objective results from a search may be the difference in making a contract. Each day, engineers work with lots of information in their systems. The importance to have good systems, and to reach the information needed quickly grows.

(Enquadramento do problema actualmente. Falar na resolução de problemas pelas TI, e nomeadamente a resolução de problemas de pesquisa, organização e recolha de informação)

## Vision

The issue of information retrieval in a society where the organization, and indexing of information itself is very useful, and even sometimes it is critical, it becomes important to develop systems and processes that eases the complication and challenges that information has.

Organizing the information in databases is one of the steps for these challenges. Organizing in a way that information systems can easily retrieve, trying to discover relevant information, related to the search pretended.

(Será este projecto uma futura solução para automatização de processos? Em que contextos irá ajudar? “Eu vejo o mundo …” ou “Eu vejo no futuro que possa existir … ”.)

## Goals

The information in the databases of the systems, tend to get bigger after time. It is very important to organize such information in a way that anyone can reach it objectively. One of the goals of this work is to purpose a technique so that the information is searchable and the results intended can be related with a query.

It is a way to help discover patterns inside the documents, and even discover related concepts in text.

One will try to help discover frequent patterns of text in a corpus of document files, and with the help of an ontology, will try to map with similar concepts, and also try to find some association of those concepts with the context of the knowledge.

(Que situações/problemas a solução proposta irá resolver? Qual o objectivo principal/secundário)

## Development context

The civil industry is no exception when the subject information appears. Like any other area, the quantity of information is growing in large scale. It is thus necessary that the ways to store knowledge, and in a form that can be understandable to an engineer, and to a machine. Text is a good form of representation of knowledge, that has search capabilities and also easy understandable.

When a civil engineer starts a project, normally works in a collaborative way with other actors, like constructors, employees, other engineers. It becomes a necessity to have a system that has all information gathered, and at same time can be scalable. This scalability also brings new challenges. How to get the information for a specific project when all projects are in the system? And if the necessity of searching documents arises? How to get the documents that are similar to the subject one search for?

This work tries to answer such questions, proposing a solution to help get the more approximate results of a query, with a technique called Association Rules.

(A solução no contexto da engenharia civil. Falar da integração dos )

## Dissertation Structure

(Estrutura da dissertação. Resumo de cada capítulo)

# State of the Art / Related Work

(Apresentar aqui alguns trabalhos desenvolvidos na área, como o antecessor deste(do Paulo); dar também uma perspectiva dos trabalhos existentes com ontologias, aprofundar alguns trabalhos que reforçam a aplicação de association Rules; e mais especificamente apresentar alguns trabalhos desenvolvidos na construção civil, dentro do tema)

## Concept Relation Quantification

(Falar com o Paulo)

## Ontology Learning

### Association rules

In semantic systems, there are several ways to reach the goals. One presented in this work and also much studied in the scientific community is the Association Rules technique. The main goal of this technique is to help making conclusions about the relation of data/words in the texts and to help build a new ontology or improving an existing one without any knowledge of the taxonomy present in it. It only relies on the original documents

## Building & Construction

# Theoretical and Technical Foundation

## Association Rules

The process of discovery of rules in data has been a subject of many researches by the community of Artificial Intelligence. In the sub-topic of Knowledge Discovery and Machine Learning, the recognition of rules in associations between items plays an important role. For an AI system be able to recognize and take some conclusions about how the information is related, therefore, Association Rules (AR) is arguably considered as one of the most important tasks in Knowledge Discovery (Marinica and Guillet, 2010), and one of the most studied in the scientific community (Agrawal and Srikant, 1994, 1994; Agrawal et al., 1993; Hoque et al., 2011; Marinica and Guillet, 2010; Paiva et al., 2013; Vo and Le, 2009; Wang et al., 2002; Zaki, 2000; Zeng et al., 2010).

Association Rules is an algorithm that identifies tendencies and relations between frequent items in a database and tries to make predictions over behaviours. Was first introduced by Agrawal et al. (1993). The problem that AR tries to address is related to the analysis of knowledge in transaction data from a collection of items. The goal is to help information analysts or automate systems making the best decisions. This is a process to improve the quality of those decisions. Additionally, association rules is a process that drives good and easy understanding to an analyst. To demonstrate the problem more clearly, the academic community refers to an example based on market basket analysis and its transactions in a large department store. In this example, the problem presented is to predict the behaviour of the clients, based on the collected data from each of the transactions. For instance, if a client buys some product A, the AR Algorithm, based on the stored transaction data of other clients, AR will give the best common behaviour that this client will have. With a set of products as the *premise*, the AR will predict a *conclusion* that the client will probably buy some product B.

|  |  |  |
| --- | --- | --- |
|  |  | (1) |
|  |  | (2) |

Two tasks are necessary to create Association Rules. First, an algorithm for frequent pattern mining in the database is applied to discover all frequent items that occur in the database, and the second task is the extraction of interesting rules among the frequent items. An AR is an implication rule in the form of equation **Erro! A origem da referência não foi encontrada.**. Two itemsets must be considered, one for the premise and other for the conclusion[[1]](#footnote-1). Itemset A represents the premise, B the conclusion, and the rule is defined by if A happens then B will likely happen also. The intersection of A with B is an empty set. On other words, the transaction A does not have anything in common with the transaction B. The itemsets that are considered in the premise can include one or more items. As for the conclusion it can hold only one.

The Association Rules can have a classification of different data types. Similarly to database attributes, they can be *Boolean* or *quantitative* (Hoque et al., 2011). Boolean association rules are the ones that hold boolean values like true or false, or 0 and 1. They are on the form of if A is true, then B is also true. Quantitative association rules are found when the items are in numerical form of some kind. One can think of when the items are intervals of numbers, like an age for instance. For the purpose of this research, because the items considered are concepts items, only Boolean association rules will be considered and designated by Association Rules.

The transformation of data into knowledge is a very challenging task. Discovering what will be interesting or not is also a good challenge for the association rules discovery process. To achieve an interesting association between two datasets there should be some kind of evaluation. In this field, some considerations and thoughts must be made when considering the evaluation of knowledge and specifically a rule. The first question that one should consider about the evaluation process is what should be evaluated and what should be considered interesting for the problem solution. One should not forget that association rules holds a wealth of information related to a data set, therefore some ways of evaluation were created to extract the best information that is more relevant. As a more broad definition, *evaluation* could be seen as the discovery of results obtained in some process, having in mind the achievement of some goals. The evaluation process is a very broad area, but this study will only be centred in the evaluation of association rules.

When the interest of evaluation has to be considered, one should start with the domain of evaluation. For instance, if the domain is construction and architecture, association rules discovered that include houses or buildings could be more interesting than computers or photography. In contrast, if the idea is to find relations to houses or photographs of buildings, then the interest on photography rises, transforming one uninteresting domain into an interesting one. Therefore, the domain is one important factor and should be carefully chosen to give the best results related to the interest considered. In Yao et al. (2006) is argued that the user can also play a crucial role. Yao et al. highlights the presence of a judge or someone who benefits from it is also important for the evaluation process. This is true as one can verify also referred by the present study, which a system is built to help this user make better evaluations. However, this approach has some drawbacks. One of them is the subjectivity of a rule. When talking on human beings, different points of view are expressed by different people, the background education can be also different, or even the geographical location can be a factor of difference when evaluating the same rule. Therefore, each of the rules is also dependable on the specific person that would participate in the evaluation process. This could be or not a problem when evaluating the rules. In literature, some approaches have been presented to evaluate the subjectivity of a rule. The subject of subjectivity of the interest of a rule is further discussed in the following sub-chapter .

To overcome the drawback of subjectivity of a rule, some objective measures have been proposed to measure a rule. The methodology to use in the evaluation process depends, as seen above in several factors. Other factors also contribute for the best evaluation as the measurement technique. Much research has been done, since the presentation of the association rules in Agrawal et al. (1993). Most of them highlight the importance of *support* and *confidence* of a rule as two metrics that can assist the discovery of interest in association rules. This study will reinforce the importance of these two, and demonstrate the existence of more metrics than the former that can be used to enrich the evaluation of a rule. The discussion of these measurement techniques will be illustrated in-depth in the next sub-sections of this chapter.

One interesting approach has been presented in Hilderman and Hamilton (2001), where the author outlines some different points of view on how to measure knowledge in general and describes some techniques to measure such interest. Some of the techniques were proposed to be used on association rules. The author discusses the measures application, in which some measures for objective and some for subjective knowledge are debated.

As illustrated above, in the previous lines, AR is a two step procedure. Before a process could discover rules of association between frequent patterns found on data, one big step has to be made. From the pre-processed corpus of data, one has to recognize frequent patterns in the concepts amongst it and transform the processed data into knowledge that could have some semantic significance and interest. This is the first step to achieve AR, and for this, there are several algorithms that propose a solution to this problem. ECLAT (Zaki, 2000), Apriori (Agrawal and Srikant, 1994) and FP-Growth (Han et al., 2004) are the most known and studied. Apriori and FP-Growth are the most used of all three.

The second step of AR discovery is the rules identification and interestingness evaluation. To achieve this one has to define first what interest is and what it finds relevant. This will be once more discussed in the following subsection 3.1.2.

To measure the interest of a rule there are several techniques that help finding the strength of a rule. Some of them will be presented in the following respective subsection 3.1.3.

This next sections are going to examine the foundations of the Association Rules with the description of the algorithm to find frequent patterns used in this study, the FP-Growth. A definition will be illustrated and an explanation of the utility of the frequent pattern procedure. Moreover, the algorithm will be explained and some discussion will be made related to it. It will be compared with the two main competitors presented earlier, and discussed what the best one or the fastest one is, and finally what is the one who develops better performance with small and big data structures.

Furthermore, some discussion around AR will be presented, namely discussion on how this technique works and what is the methodology used. Some other questions will be answered, like what is a rule or how can one define a rule. Subsequently the metrics of a rule will be debated. How can a rule be measured, what metrics are known, what metric is the best, what makes a strong rule and what are the metrics that are most used will be some of the questions argued.

All this questions will be answered by the author of the present work along with some discussion around some other studies in the field of Association Rules and FP-Growth.

### Frequent Pattern Growth

Before rules of association can be found, the database must be mined to see which of the items are frequent. There are several processes in the academic community for this purpose who, given a set of database transactions can search it and return all the frequent item above some kind of measure to prove that represents the frequency of each item. Apriori, Eclat, FP-Growth are the ones that are most used and discussed by researchers. Han et al. (2004) is recognized as a first introduction of the FP-Growth approach. It compares FP-Growth with Apriori, one of the initial and most used processes.

There are some characteristics that an algorithm should have to be classified as a good one. Namely time performance, usability in large databases and small databases, scalability, etc. In the next sub sections, the arguments are in favour of FP-Growth, in which the author of the present work identifies, based on the research, as being the best for the present case.

#### FP-Growth - A Definition

FP-Growth stands for Frequent Pattern Growth, it represents an algorithm to discover frequent patterns in data and specifically used in text mining. This algorithm is currently one of the fastest ones to mine association rules. It can also be defined as the first step in the path of item recognition that appears frequently. These items are called Frequent Patterns, meaning some text that appears in the database of transactions and are considered frequent above some minimum threshold value. This value is chosen by the engineer handling this process. It is more or less an arbitrary choice based on try and error method. More work can be done in this step to improve results, such that an artificial intelligence process could find a method to dynamic choose the best value for the intended use of the algorithm.

Before FP-Growth the processes to discover frequent patterns in databases of text were mainly Apriori-like based algorithms. Such processes are known to be very costly in large databases. Its times to search will exponential grow as the database will also grow. On the other side stands FP-Growth, a *divide-and-conquer method* (Han et al., 2004). It is based in a prefix tree representation, called FP-Tree. This tree holds the frequent patterns found in the transaction database. With the divide and conquer method this can be seen as a recursive elimination process. It will separate the frequent items from the ones that are not frequent inside a database.

Due to the popularity, effectiveness and performance of this algorithm, it was much appreciated in many investigations in the academic and scientific community. Also many changes proposals to the original were studied and presented. For example, in Wang et al. (2002), the author proposes improvements and upgrades for the algorithm. One of these proposals is the Top Down FP-Growth algorithm. This work’s author debates a different process to search frequent patterns. It searches the FP-Tree from the top to the bottom and not generating conditional FP-Trees to each item. This method processes the nodes of the tree at upper levels before processing the ones on the lower levels. This is different from original FP-Growth, in which it mines the tree from bottom up, from the item to its prefixes, and creating several conditional trees for each item.

Korczak and Skrzypczak (2012) illustrate an example of discovering customer frequent patterns in an online store with the help of FP-Growth to discover association rules between the transactions of the customers. In other project, Bonchi and Goethals (2004) are inspired by the known small Japanese bonsai tree and tries to apply its broad concept in the FP-Tree of the FP-Growth algorithm. This study examines the reduction of the tree by a technique that is based on pruning specific “leaves” (nodes) resulting in smaller compressed trees.

Another study related to this subject is the one presented in Zeng et al. (2010). In this paper, the discussion presented is a process to weight association rules based on an FP-tree. It proposes a new method called FP-Weighted Association Rules (FP-WAR) where outlines the importance of getting a technique to weight association rules and give them different *interestingness*.

#### FP-Growth Algorithm

The FP-Growth algorithm is made in two steps. The first is an elimination phase scheme and building of an FP-Tree, and the second step is a frequent itemset generation.

Table 3‑1 – Transaction table for frequent items in database

|  |  |
| --- | --- |
| TID | Items |
| 1 | architect, designer |
| 2 | designer, engineer, analyst |
| 3 | architect, engineer, analyst, professor |
| 4 | architect, analyst, professor |
| 5 | architect, designer, engineer |
| 6 | architect, designer, engineer, analyst |
| 7 | architect |
| 8 | architect, designer, engineer |
| 9 | architect, designer, analyst |
| 10 | designer, engineer, professor |

**Step 1 – Infrequent items elimination**

The elimination scheme is where the initial data is mined to separate the frequent from the non frequent items in the database. It uses a recursion process to make a kind of elimination scheme. It compresses the data set by determining the frequent items and deleting all that are not frequent. These infrequent elements are the ones that are found mainly alone in the database, this means, that such frequency is below the above threshold level. On the other side, the frequent items that are above this level are called frequent items, and are the ones that it will be kept, and are the base for the next step.

As already referred, the deleting process is based on a user-defined minimum offset, called support, in which below it, all items are eliminated, and above it the items are the ones to be stored. This value along others is explained in detail in sub section 3.1.2 below when the discussion will be around how to measure an item. The result of this process is a modified set of transactions with only frequent sets of single-items. In the end, the frequent items are ordered for better organization search purpose. presents an example of a resulting set of transactions after this elimination and ordering process be made. Each line of the table represents a transaction from the database. The resulting items are the ones found in the database that matches the requirements of a threshold value above the user defined value. These items are the ones considered frequent in our initial data.

Following the elimination scheme and frequent itemset filtering, a tree has to be built, the so called FP-tree. This tree is a rooted acyclic graph with vertices not labelled and a root node null valued. It is constructed with transaction scans like the ones on Table 3‑1, one transaction at a time. The main idea is to map the graph such that a new path for each unique transaction will be drawn. Each node represents each frequent item found earlier. If the search discovers a common prefix on the item set, it will overlap and remove it, and if a suffix exists, creates a new node in the graph and connect to its previous item. If this common prefix is shared along two transactions, these are merged into the corresponding node. Each node holds a counter. This counter represents the frequency of the node in the respective path and always starts with the value of 1 in each node creation. When a transaction shares a node along the same path, the counter is then incremented, and it goes to the next transaction. This will ensure that each frequent item only needs one path for each item in the tree. The chances of which the common prefixes can be shared are higher if the frequent items have been sorted by its frequency, from top to bottom order.

Figure 3.1 – An FP-Tree example for the items in the transaction .

Example: let’s suppose that there is a database that after applying the elimination and ordering task of FP-Growth presents the following set of items (3), and the previous transaction with all 10 transactions of this database scan:

|  |  |  |
| --- | --- | --- |
|  |  | (3) |

**Building the FP-Tree**

The algorithm to build the FP-tree will have to deal with each of the transactions, one at a time. First it should be created a node to represent a root with the value of *null*. For the first transaction TID1, the first item is *architect*. This item will create a new node in the tree with the value of *architect* and a counter associated to it with the initial value of 1. The second item in the TID1 is *designer*, as this is also a new one, a node should be created in the next level of *architect* node with counter equal to 1. Figure 3.2a) represents the situation after TID1 where one can see the FP-tree constructed until this moment. For TID2 the items *designer, engineer and analyst* are the ones to consider. As the first one is *designer* and in the current tree there is no first level node with such designation, a new node should be created a connected to *null* node for the item *designer*. As this is a new node its counter is set to 1. Then the node for engineer must be created and connected to designer with the counter at 1. The next item on the transaction is *analyst* that it is also a new one in the path, so a node must be created with its name and the counter equal to 1. The moment at the end of this transaction TID2 is represented in the b). One can see that there is two individual paths for the transactions, both sharing a node with the same item: *designer*. In this case both should be linked to recognize this situation and further evaluation. The dotted line represents this linkage. TID3 includes items *architect, engineer, analyst and professor*. One can easily see that the first item of this transaction already is connected to the *null* node. In this case, there is no need to create a new node that would be repeated, instead the respective counter should be incremented by 1, totalling now 2, that represents the two paths that starts with *architect*. For *engineer*, *analyst* and *professor*, the procedure should be similar for new nodes in the same path. As *engineer* and *analyst* already exist in the tree, they should be linked with its equals also in the same way *designer* was above. An illustration of this transaction is Figure 3.2c). One can see that three different paths exists if counting the last leaves of the branches or if totalling the sum of the nodes and counters connected to *null* node.

Figure 3.2 – The three initial trees at the end of the first three transactions

**a) TID = 1**

**b) TID = 2**

**c) TID = 3**

This process shall continue through all transactions until all transactions table lines be evaluated for the actual process. For this example the table has 10 transactions, which are all represented in above. From this figure it can be seen all the paths from the transaction table. To be noted that node *architect* is the one that starts most of the paths of the itemsets, exactly 8, making him the most frequent item in the database. As we can also infer from the figure, is that *designer* is the second most frequent, but it alternates in its position in the paths, five in the first position and 5 in the second position. It can also be noted in the tree that all items nodes with the same designation in the different paths are linked to each others.

The main purpose of this step is a filtering and organizing step with the objective to facilitate the search of frequent items. With this kind of structure the speed of search will decrease significantly.

**Step 2 – Frequent Itemset Generation**

The next step in FP-Growth algorithm, as noted, is the evaluation and recognition of the most frequent items. It is an extraction process that is called **Frequent Itemset Generation**. The main idea of this technique is to extract frequent items from the earlier built FP-tree. From a Bottom up perspective, the technique will individually process each ending bottom node and separates in an exclusively separate tree which is called **prefix path sub-tree**.

Figure 3.3 – Prefix sub-paths for all frequent items

**a) Prefix path sub-tree for *professor***

**b) Prefix path sub-tree for *designer***

**c) Prefix path sub-tree for *engineer***

**d) Prefix path sub-tree for *architect***

**e) Prefix path sub-tree for *analyst***

Find an example of this trees in where each of the items in the database will have an exclusive prefix path sub-tree. It is the *divide and conquer* method, separating each sub-tree individually for faster performance of the main tree. The paths included in each sub-tree are the ones that have the respective item as a leave node (ending bottom node). Therefore, for a frequent item X and its ancestor Y and Z from an FP-tree, the resulting sub-tree will be used to extract itemsets ending in X, subsequently will extract the ones ending in YX, and after it the ones ZYX, continuing traversing through the path and being processed recursively until it analyses all paths and reaches the most top node, also called null node.

Table 3‑2 – Paths table for frequent items

|  |  |  |
| --- | --- | --- |
| Item | # | Paths |
| Architect | 1 | architect |
| Professor | 1 | architect, engineer, analyst, professor |
| 2 | architect, analyst, professor |
| 3 | designer, engineer, professor |
| Designer | 1 | architect, designer |
| 2 | designer |
| Engineer | 1 | architect, designer, engineer |
| 2 | architect, engineer |
| 3 | designer, engineer |
| Analyst | 1 | designer, engineer, analyst |
| 2 | architect, engineer, analyst |
| 3 | architect, analyst |
| 4 | architect, designer, engineer, analyst |
| 5 | architect, designer, analyst |

Using the previous example, a) to e) represents the prefix sub-path trees for itemset S. For each of the individual items in S, one sub-path tree was divided for further processing. In a), the prefix path sub-tree for item *professor* presents 3 paths described in . Consequently, the *divide and conquer* approach makes the problem easier to evaluate.

As observed in the first step, for a node be considered frequent it has to hold a support threshold value (minSup = 2 in previous example). That was a requirement to search the items in the database and eliminate the ones that did not have at least another equal item in it. With this minimum support in mind, one has to traverse from the bottom to find each frequent items. For this task, if one wants to know if X is a frequent item, it has to follow the dotted lines in the prefix sub-tree and sum the counters associated with item X, and thus calculating the support for X, denoted by (Vo and Le, 2009).

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

Equation **(4)** represents the mathematical expression to consider an item X a frequent item in the database. If this situation is true, X can be extracted as a frequent item and it can be found frequent items ending in X. For that a table as has to be considered holding all paths for each of the extracted frequent items.

In previous the example, considering the item *professor*, one can easily sees the 3 paths for it. They are {architect:8, engineer:1, analyst:1}. {architect:8, analyst:1} and {designer:2, engineer:2}, these all lead to an ending node *professor*. In case of the third path, the set {designer, engineer} appears twice in database, however with *professor*, they only appear once. Similarly, architect in the other two paths is shared among them with a support of , appearing only twice together with *professor*. Therefore, to evaluate the set that appears together with *professor*, the nodes from the corresponding prefix path sub tree have to be updated as follows:

S1={architect:1, engineer:1, analyst:1}, S2={architect:1, analyst:1} and S3={designer:1, engineer:1}. Furthermore, as *designer* has, it's not considered FI with *professor*. Thus S3={engineer:1}.

Figure 3.4 – Conditional FP-Tree for item *professor*

These three prefix paths of *professor* {S1, S2, S3} constitute the sub pattern-base called, according to Han et al. (2004), conditional pattern base, consequently, an FP-Tree based in this is called **conditional FP-tree**. A visual representation of the conditional FP-Tree for item *professor* is illustrated in Figure 3.4 – Conditional FP-Tree for item *professor*Figure 3.4. (FIGURE 3.4)

Table 3‑3 – Frequent itemsets discovered for all items

|  |  |
| --- | --- |
| Item | Frequent Itemsets discovered |
| professor | {professor}, {analyst, professor}, {architect, analyst, professor}, {engineer, professor}, {architect, professor} |
| analyst | {analyst}, {engineer, analyst}, {designer, engineer, analyst}, {architect, engineer, analyst}, {designer, analyst}, {architect, designer, analyst}, {architect, analyst}, |
| engineer | {engineer}, {designer, engineer}, {architect, designer, engineer}, {architect, engineer} |
| designer | {designer}, {architect, designer} |
| architect | {architect} |

includes all frequent items found in this example, considering all items in database.

#### FP-Growth versus APRIORI and ECLAT

Although FP-Growth is a very efficient algorithm to find frequent patterns in databases, it is not the only one, nor even the first one to appear. In the last years, much study was conducted in frequent patterns recognition in data mining subjects of Artificial Intelligence area. Since the presentation of the problem of association rule mining in (Agrawal et al., 1993), many algorithms appeared in researches all claiming to be the best for some reason. For instance, APRIORI, that was one of the pioneers to address this situation and introduced in Agrawal and Srikant (1994), is a starting point for many studies in frequent pattern discovery. In this research, Agrawal et al. defines this algorithm as a procedure for candidate generation. These candidates are used to construct other candidates in the next level and frequent itemsets. One of the main problems recognized in Apriori by the scientific community (Han et al., 2004; Zaki, 2000) is the number of scans it uses to generate the frequent items from the candidate retrieved from the database. It performs as many searches in the database as the maximum element number in an itemset of candidates. Hence, as bigger the candidate sets are, lower is the performance of the algorithm. It starts to be even worse when the size of the database tends to be large, although it could discover the frequent items, it is a little boring to repeatedly search a large set of candidates by pattern matching. In the meantime several other attempts tried to improve Apriori algorithm. Some examples are MSApriori (Liu et al., 1999), A-Close (Pasquier et al., 1999), Apriori-Inverse (Koh and Rountree, 2005), UApriori (Metanat Hooshsadat et al., 2012) and many other Apriori-like based algorithms.

Similarly, ECLAT is another studied algorithm to find frequent itemsets in databases. ECLAT stands for Equivalence CLass Transformation. This algorithm was introduced in (Zaki, 2000) as one that would improve the performance problems of Apriori-based algorithms, like minimization of I/O costs reducing the number of database scans or event the reduction of the computation costs with more efficiently search procedures. ECLAT needs just a reduced number of scans in the database and no hash trees whatsoever as it generates frequent itemsets by only simple intersection operations. It can even handle support values lower than, for instance, Apriori in large datasets.

One of the advantages of FP-Growth, when comparing with the competitors is that it does not create huge amount of frequent itemsets and a small database of transactions. It only needs one scan on the database, along with a minimum support threshold to scan it and discover frequent itemsets. As observed in the previous lines, Apriori and most Apriori derived algorithms are candidate set generation algorithms, on the contrary FP-Growth is not. It does not need to make such a costly operation to generate frequent items, in contrast, it uses mining operations of count accumulation (frequency count) and prefix path count adjustment. These are less costly than candidate set generation and pattern matching operations.

Borgelt (2005) presents an interesting study on these three algorithms, Apriori, ECLAT and FP-Growth. It argues that the implementation of the process of frequent discovery in FP-Growth “*clearly outperforms Apriori and ECLAT*”. Even after the previous were improved and optimized.

In brief, one can see that FP-Growth have more strengths than weaknesses, when comparing to others algortithms. It recognizes frequent patterns in data, and needs less time to give the results than its competitors.

### Association Rules Measurement

As presented before, Association Rules algorithm recognizes associations in frequent patterns resulting from a frequent pattern recognition algorithm like FP-Growth or Apriori. The following step is responsible to evaluate the rules in a way that it will show interest to the subject. Several publications have appeared in the recent years identifying ways to measure the interest in an association rule. In the following lines, it will be discussed how association rules could be measured. It is also discussed what should be thought as interesting to retrieve from the rules, and the ways to do it., in the form of subjective and objective measures.

#### Subjectivity and objectivity

To be able to measure interest in the knowledge discovered, two types of measurement of the interest of a rule are identified: Subjective and objective measures. Mackie (1977) presents a study where he describes that the subjectivity in evaluation is very common when the evaluation goals are objects, actions or events. The objectivity is used in the measures themselves and their implementations. The subjectivity depends in great factor on the person that is considering the subject. As explained before, it depends in factors like the location or the background. Some other studies also discuss the subjectivity and objectivity of a measure. Silberschatz and Tuzhilin (1995) propose a classification for interestingness measurement of a rule. As one could see in where this classification is illustrated, it is argued by the author that the interest should be divided in *Objectivity* and *Subjectivity* measures.



Figure 3.5 – Interestingness measures types tree (adapted from (Silberschatz and Tuzhilin, 1995))

Some of the challenges to evaluate the subjectivity of knowledge interestingness, and in this case, a rule, were already discussed in the beginning of this chapter. But it is important to go deeper in this subject. Silberschatz and Tuzhilin (1995) points out in their study that it is important to measure the subjectivity of a measure. In the subjectivity side of the classification tree in , the author divides this subjectivity of a rule in two concepts, *Unexpectedness* and *Actionability*. The first concept represents the value of some unexpectedness in a rule when knowledge is discovered. If one could discover a rule that it would not expect, that rule would be interesting. Of course, some knowledge expected, is knowledge that is already known, and thus, not interesting for the user. The second concept of subjectivity, actionability, represents the usability that a rule could have. In other words, it is the capacity of a rule to be used in an interesting way by its user. One example of these concepts applied to a rule that is presented by Gonçalves (2005) and could help explain these kind of subjective measures in a rule is the association between dippers and beer in a big department store. This example explains that when the transactions are made by young couples on a Thursday, this association is detected. The company analysts would think that the buying of beer would just be associated with the buying of appetizers or barbecue meat and other alcohol drinks. Surprisingly, when association rules are discovered, this unexpected knowledge rises to the edge. This is the perfect example of an unexpected and actionable rule, and as a result, for now on, on Thursdays, the department store can use this extraordinary new and unexpected discovered knowledge to move the dippers and the beer closer to each other, so the sales could go higher.

Although these two concepts are independent of each other, they can be combined to strengthen even more one rule. Regularly the unexpected rules are also rules that are useful. Similarly, the actionability rules, the ones that an engineer can do something useful with them, are also rules not expected to appear. If one thinks a little deeper, this makes sense. If the object of association rules were to result knowledge that was already known, what would be the point?

On the other side of the interestingness tree, are the objectivity measures. These measures statistically identify the strength of the association rules. It is important to know some characteristics that one would want in a measure. In this matters, Tan et al. (2002) describes a list of several measures found in the literature and discuss some properties of a measure. In this work are presented some properties that the author defends that should be desirable and applied to the measure operation of association rules. Three properties are presented in the work as the more relevant, the first one states that if one has concepts A and B that are statistically independent, then the measure is equal to 0. This means that if a rule does not find any relation between the concepts these are not related. The second property presented states that a measure increases with the support of a rule, when P(A) and P(B) remains the same. And the third property presented as the considered desirable for the authors, describes that a measure decreases with P(A) (or P(B)) when the other parameters remain unchanged, namely the support, P(B), or P(A) respectively. Several more properties are presented in this work and the author examines each of the measures against each property. This is a good way to justify each of the measure considered.

#### The measures

On the next lines, based on the interestingness tree presented in , the measures will be discussed and presented. All of them will be identified and discussed, namely support, confidence or conviction and lift, or even gain, laplace and ps. The existence of some other measures will also be presented and discussed.

**Support and confidence**

The majority of the works studied about objective measures of association rules, take advantage of *support* and *confidence*. Hoque et al. (2011) which presents a document on association rules consider these two measures, presented also in , to find interest in association rules. Furthermore, Azevedo et al. (2005), Bhujade and Janwe (2011), Brin et al. (1997), Kumar and Chadha (2012) and Spruit (2007) are examples of researches where support and confidence are used to measure rules of association. They all use at least these two measures to extract knowledge and to take some conclusions over the rules. Additionally, Bayardo and Agrawal (1999) also considers the use of these two measures as a way to reduce the rules to the most interesting. The former and Azevedo and Jorge (2007), Gonçalves (2005) and Tan et al. (2002) also present some definitions of each one of the measures. The rule *support* (also *frequency, coverage*) is equal to the number of occurrences in the corpus of data information where both concept A and concept B evaluates to true. This is presented as sup(A⇒B) (also defined in some literature as σ(A⇒B)). In other words, the support of a set of items, that is a statistic metric, is defined as a transaction percentage from a database where these items are included. As this is a statistic measure, the values are between . Higher the value, the more frequent are the concepts in the database. The definition for support is represented in the following equation (5).

|  |  |  |
| --- | --- | --- |
|  |  | (5) |

Furthermore, *confidence* represents an estimation of the probability of observation Concept B given Concept A. When a rule is received, one can immediately classify the relationship of the corresponding concepts. The expression to calculate confidence is given in Equation and the result values, as this is also a statistic measure, are enclosed in . One can also identify that the interest rises also with confidence results.

|  |  |  |
| --- | --- | --- |
|  |  | (6) |

These measures, although, alone give little information. To get the real interesting rules, one has to consider two additional parameters, *minsup* and *minconf*. These two parameters propose a lower limit on the interest of a rule. For instance, a rule can have a support value of 20%, however, if the defined minsup is 50% this rule is considered uninteresting. Bayardo and Agrawal (1999) argue the definition of some these borders. Their objective is to propose the discovering of the most interesting rules using this border defined by minsup and minconf.

In the time of this research the author did not find an alternative for an automated process to choose this limits, as a result these values have to be an arbitrary choice. And two problems arise immediately when choosing these values. If the values chosen are too low, it could result in too much rules to analyse, and in redundant information, in contrast, if the values chosen are too high, the interest of the rules could be low as some of the knowledge is already known, resulting in expected and/or useless information. These values have to be wisely chosen, and in a balanced way, so that could select some interest, and at same time select the most interesting knowledge. This choice could be done with little adjustments until the results are considered a good choice.

**Conviction and Lift**

Although support and confidence can give a real good and trustful interest measure results, they sometimes are not enough, as a result some other measures were studied and used in the scientific community. *Conviction* and *Lift* are other two measures that were proposed to complement the former, and that are also commonly used to strengthen the conclusions obtained from confidence and support measures. Also statically measures, these two depend on their values to be calculated.

Conviction is a implication measure that quantifies the value of the implication, it is represented as A⇒B, meaning that the direction of the rule is important for the interest measurement, hence A⇒B ≠ B⇒A. Conviction measure has some very interesting properties, such as if its value is equal to 1, this means that the concepts are considered totally independent from each other. Other property is that this measure considers the value of the antecedent as also the value for the consequent to calculate its value. Other interesting property of conviction is on rules with 100% confidence value, meaning where the antecedent always appears with the consequent, these rules will have the value equal to ∞. To achieve the most interesting rules one can think of as higher the value of Conviction, higher is the interest of that rule. The values of Conviction are included in . Conviction can be defined mathematically as presented in Equation (7) or can be also presented as Equation (8) dependant of confidence measure.

|  |  |  |
| --- | --- | --- |
|  |  | (7) |
|  |  | (8) |

In contrast to conviction that measures implication, Lift[[2]](#footnote-2), (can also be found in literature as *Interest* in Brin et al., (1997) or as strength in Dhar and Tuzhulin (1993)) is a measure that quantifies the co-occurrence of a rule. Lift is not an implication measure, it means it is symmetric in relation to the antecedent and consequent, hence Lift(A⇒B) = Lift(B⇒A), in other words it measures how far from independence are concepts A and B. Lift is defined as a measure to boost (“lift”) the confidence of a rule, this suggests an improvement of the trust of results of rule confidence. Similarly to conviction, if its value is 1 it means they are total independent without any kind of interesting relation, and as far from 1 and as higher the value is, higher will be the interest of the rule and more relation can be found on them. The set of values of this measure are included in . Lift is defined by the following Equation or also defined in Equation where one can see the dependency from confidence measure.

|  |  |  |
| --- | --- | --- |
|  |  | () |
|  |  | (10) |

As can be easily seen by both equations from conviction and lift there is a relation to confidence measure. Therefore, these measures can be understood as measures to help improve or strengthen the trust on confidence results where the confidence itself would not be enough to make the conclusions and find relevant knowledge in the association rules. In the case of Lift, the measure is better for rules with lower support.

**Gain, Laplace and PS**

In some more measures are illustrated. As one can see, they are presented as Laplace, PS and Gain. These three are also measures dependable of support.

Laplace is a classifier that is one of the additional measures considered in this research. It can be considered as a confidence estimator that is function of support, and as low as support is, lower is the interest in the rule considered. Laplace is normally used to rank rules by class. The range of values are in . Its mathematical definition is the following Equation (11). The constant *k* represents the number of classes defined when one is defining the respective classification model. Its value is always higher than 1.

|  |  |  |
| --- | --- | --- |
|  |  | (11) |

Another one of these measures is Gain. This is an optimization measure presented by Fukuda et al. (1996) and discussed by Bayardo and Agrawal (1999), and by Brin et al. (2003) as a proposal to solve the optimized gain rules problem. It is defined also as a function of support and given by the following Equation (12). The parameter is defined as a constant fraction with values between 0 and 1. Additionally, if one wants to decrease the subtractive term, it can be only done by decreasing the support of the antecedent. When this happens, the confidence value becomes higher.

|  |  |  |
| --- | --- | --- |
|  |  | (12) |

The last of these three measures presented is *PS*. This measure receives its name from their creators, Piatetsky-Shapiro (1991)[[3]](#footnote-3). It was originally used to classify rules, and later adopted by association rules. This measure is a boost to the support measure. As it gets a value in the range . If its value is equal to 0 it means that A and B are independent. A value below 0 represents a negative dependency and if the value is higher than 0 it is called positive dependent. Higher values represent more interest in the association rules. The definition for PS is presented in the following equation (13).

|  |  |  |
| --- | --- | --- |
|  |  | (13) |

**Other measures**

All the values that are illustrated should be enough to classify any association rule extracted, although, in the academic community several other measures were studied over the years. For instance, the interesting work in Tan et al. (2002) evaluates a list of 21 measures for association patterns, where the measures studied in the present research are also considered and evaluated. Some others like gini, entropy gain and chi-squared are also discussed in Bayardo and Agrawal (1999). Further research can be made in the direction of more measures to improve the association rules process reliability.

## Vector Space Model

In Text Mining, a Vector Space Model (VSM) is an Information Retrieval statistical model that tries to make the assumption that each document is represented by a point in space in a group of documents. As more near the points are, bigger is the semantic similarity and as more apart the points are, less representative in the semantics they are(Turney et al., 2010).

The VSM is a form to explain to computer systems the semantics of human language. It was created for the SMART information retrieval system by its developer Gerard Salton and his team. (Salton, 1971)

VSM has several properties, one of which is that given a corpus it will extract knowledge automatically. The majority of today search engines use VSM as a model because of its good performance in preparing the raw data to measure the similarity between documents, phrases and words (Manning et al., 2008). Queries made by the engines have a good performance presenting relevant results to the query author. Some of the most known algorithms for semantic relatedness(Pantel and Lin, 2002; Rapp, 2003; Turney et al., 2003) and semantic relation similarity (Lin and Pantel, 2001; Nakov and Hearst, 2008; Turney, 2006) also use VSMs as a base technology for preparation of the data.

There are some hypotheses that VSM tries to answer, they all begin from the main one, the *statistical semantic hypothesis*, that states that if statistical patterns are used on human word syntactic formation and usage of natural language terms, the possibility to understand the meaning of human speech is real.(Turney et al., 2010) The above hypothesis is the converging point of the following ones: bag of words, distributional, extended distributional and latent relation. In the following lines, the author of the present work will give a brief explanation of each of them.

* **Bag of words hypothesis:** By representing the documents on the corpus and the query as a bag (or collection) of words, one can estimate the relevance of these documents to a query. This can be explained as the word frequency that exists in the documents tends to represent the document relevance faced to a query. Each bag of words can be represented by a Term-Document Matrix. Salton et al. (Salton et al., 1975) wrote that this hypothesis is the foundation of VSM application in information retrieval systems; The authors of this hypothesis believes that each column vector of this matrix represents in some way a subject or meaning of the document. (Salton et al., 1975)
* **Distributional hypothesis:** When the subject is similar contexts, the distributional hypothesis says that the words in those contexts are also similar in their meanings. (Harris) The data is organized in a Word-Concept Matrix. When one wants to measure the word similarity, this hypothesis is the reason for the application of the VSM;
* **Extended distributional hypothesis:** This hypothesis was proposed by Lin and Pantel (Lin and Pantel, 2001), the co-occurence of patterns in similar pairs, will lead to similar meanings. The co-occorence of X:Y similar pairs is a tendency of patterns like “X solves Y” or “Y is solved by X”. When this happens one can be lead to think that these kind of patterns present the similar meanings; (Lin and Pantel, 2001) The representation of this pairs results in a pair-pattern matrix.
* **Latent relation hypothesis:** The last hypothesis is the inverse of the extended distributional hypothesis described before. It covers the pairs of words, when these co-occur in similar patterns. In this case one can say they have similar meaning.(Turney, 2006)

### Term Weighting – The TF-IDF

When dealing with large raw data, these can be represented by vectors in a matrix, the Term-Document Matrix. This matrix is prepared in such way that the terms are arranged in row vectors and the documents are arranged in the column vectors. Each entry in the matrix corresponds to a weight of each term in a document. This process orders the terms in a document by their relevance in each document and in corpus by a tf-idf (term frequency – inverse document frequency) weighting normalized scheme. This scheme is presented by the following equation (14):

|  |  |  |
| --- | --- | --- |
|  |  | (14) |

Where represents the term frequency of term i in document j and represents the number of documents that contains term i. The result is the following matrix (15) with the weight or relevance of each term.

|  |  |  |
| --- | --- | --- |
|  |  | (15) |

By this weighting process, the system may know the relevance of each term in the context and which one is more or less representative.

This form of representation is called a bag or multiset, and supports the bag of words hypothesis discussed earlier. This way, one can discover the tendency of the proximity of a document to a subject, by this frequency of words in the document. Salton et al (Salton et al., 1975) states that this hypothesis is the foundation of VSM application in Information Retrieval.(Salton et al., 1975)

The VSM is not the only way to represent text, as seen the several hypothesis represent more ways to represent and weight relevance of the terms. But this is not exclusively, for example, proposes another form of weighting the terms and documents in a corpora. This point of view, which is called TWEAK, uses labels to learn the terms weight related to its importance in the subject as a parametric function, where the model parameters are learned from the labelled data. (Yih, 2009). Instead of being an independent weighting scheme like tf-idf, that it does not take consideration the previous analysis or other kind of past similarity calculus, this TWEAK is dependant of the previous analysis as this considers the model parameters in the evaluation. Meaning that the previous labelling and classifying of the terms in the corpus are included in the next weighting, making this process influenced for the actual subject of the text data.

## Similarity Measure in Information Retrieval

(A importância da Similarity Measure na área de Information Retrieval, talvez falar de algumas medidas que existam. A Semelhança entre o quê? Documentos e queries! Word similarity; Document similarity; Context similarity)

Before one can understand what to measure, it is important to understand what similarity is, and what is it role in Information Retrieval (IR). Similarity is the state or the fact of being similar.(Oxford University, 2006). To understand what Similarity means, it is important to understand that each word or concept has a(many) meaning(s)/subject(s) that can be related to. How similar is each word to a subject? Lin even presents a Similarity Theorem to explain it:

“*The similarity between A and B is measured by the ratio between the amount of information needed to state the commonality of A and B and the information needed to fully describe what A and B are."* (Lin, 1998) In other words, a similarity of two words is a quantification of their differences and similarities. It is a measure of how much their meanings are close. How much information do they share, and how much information they do not.

Even in words with similar meaning, similarity measures are important to find what the word that best fits in a particular context is. Or just to know what the word that is more similar to other is.

In the previous chapter, the author presented an approach on how the raw data can be organized for further evaluation. For it there are some ways to do it, several take vector models as starting point. The VSM is one of the best known and applied method. As VSM states, to better understand the meaning of a word or concept, there must be a measure in the semantic relations of each word, given a set of documents. One must know how similar each word is, therefore it can try to figure out what is the best approximation to the main subject of each document, and thus give the possibility for the machine understand its meaning. For example, given the words *Architect*, *Engineer* and *house*, how can they be related to each other? The reader will obviously know that an *Architect* is more related to a *house* than is an *Engineer*. For the reader is easy to know the meaning but for an AI system, is it? How can it understand the relation to each word? This is called a *Similarity Measure*, on other words, it is a way to measure the semantic relation of each of the words or concepts, and see if they are more related to ones, than others. Several paths are possible to consider and achieve the previous.

Similarity measure can be used for more than similarity of words and concepts. One of the most known applications of similarity is between a user query and some document pages like a web search engine (“Google.com,” 2013) But this is not an exclusively use of similarities, also for instance, one can measure a document similarity between scientific papers, or measure similarity of unstructured data texts so a context or domain can be found. It can be used to aid ontology construction, fortify relations between concepts, make ontology dynamic and capable of learning.

To start a similarity measurement some initial thoughts must be made. What is it going to be measured? The corpora documents? Words? Vectors? The type of the initial corpora is important. Other thoughts must be made on if the domain is known, is the data already structured? Is there any ontology to support the process or any previous similarity measurement to help? Are their semantic similarities important? What is the similarity measure process to use?

There are much similarity measures in the scientific community that are being discussed. One of the most known is the Cosine Similarity Algorithm.

### Cosine Similarity Measure Algorithm

# Building & Construction Domain Ontology

(Falar das ontologias, o que são, para que servem, como funcionam.

Falar da importância das ontologias.

Falar da importância das ontologias no domínio da construção civil, e especificamente na construção de edifícios.)

## Ontology

The term “Ontology” origins from early 18th century, from the modern Latin *ontologia*, from the Greek work *onto*, which means “being”, plus *logia* which means study, science, theory. (Oxford University, 2006). It is, thus, a sub domain of philosophy that studies the being, to be. In its origin, is the study of the nature of being, existence and the way this being relates with others. Also represents the categories of the being and such relations.

In information systems, the ontology is the study of the representation of knowledge. It is a form of definition and organization of the knowledge domain. Ontology can be defined as a structure or a form of data organization. For a better understanding of this knowledge, an ontology is divided in *classes*. These classes present *relations* between them. Each class includes a set of *concepts* and their relations. The ontology is this way, a structured representation of a data set of knowledge that aims for a better understanding of the data presented.

In IT systems, W3C created a standard to define a representation and organization of a domain that can be “easily” read by a human. This standard is named OWL, which means Web Ontology Language. (W3C, 2004)

One of the main challenges that a set of raw text data presents is the difficult to translate it to a language that computer might better understand and even further process. This challenge is somewhat solved with the OWL. This language appeared to facilitate the interface between a raw data, unstructured that is not easily read and a structured data that human eye could make conclusions about it.

### Construction Methodology

#### Types of Ontologies.

#### OWL Ontology Language (and RDF)

### Ontologic enrichment Dynamics

#### Update relations

#### Creation of new concepts and relations

#### Relation Weighting

### The E-COGNOS Project – Ontology in Building and Construction

# Design and Implementation

This chapter will illustrate all the information related to the implementation for the solution of the problem presented in this research. From the modelling of the problem, design, requirement analysis, and building until the development of a final solution. Based on the model proposal of the problem, all the initial requirements were fulfilled in the technical solution. This solution will be illustrated in the following lines with a description and explanation of the technologies adopted.

## Tools and Technologies

This project could be developed in several ways, however, the author chose to use the following technologies and tools based on the initial requirements of the research. These tools and technologies were available free and were competent enough to develop this client-server product (). To model the problem, it was used UML with Visual Paradigm to design and model all respective diagrams. This software is a good design tool to develop projects in UML as it can make all diagrams necessary, namely the diagrams for Classes, Use Cases and Sequence among others.



**XPATH/XSLT**



Figure 5.1 - Tools and Technologies adopted

To model the database necessary to hold all data from the project the technology chosen was SQL with the use of the MySQL Workbench tool. This is a good tool to use with database modelling. This tool allows the creation of the Entity Relation Diagram that afterwards was translated into SQL with all the code necessary for the database building. The database runs on top of Apache server through XAMPP software package. This package also gives the possibility to maintain the databases through PHPMyAdmin web page.

Rapidminer was the tool used to make all the previous steps necessary to discover association rules. This is a software tool that is indicated to use in machine learning, data mining, text mining, predictive analytics and business analytics processes. All the pre-processing of the unstructured data was developed in this tool. With this powerful tool, it was also used a script in Groovy[[4]](#footnote-4), an agile and dynamic language for the JVM. to prepare the data for export. This script language integrates well with all existing Java classes and libraries in the Rapidminer because it compiles straight to Java byte code so you can use it anywhere you can use Java. The last component used of this tool is an included API that was used to execute the process modelled, (process represented in XML), underneath in the server, this was used to make the interface between rapidminer and the java code.

The core of this project was developed in Java in the SE Platform. The IDE used to make all the coding to hold this product is the NetBeans IDE. This IDE is an open tool that is very complete. It allows the coding of Java, compiling of the code directly to JVM and also includes a debbuger to check all the possible errors in the code or in the system run itself. All the necessary processes and methods to communicate with the Front-End are in JavaBeans interfaces or through XML data files that is serialized with JDOM on the side of the server and in the client with XPATH/XSLT for presentation purposes. The communication with the database that holds the main model is made through JDBC connections ().

Protegè was the tool to deal with the ontology as a user point of view. It allows opening, visualizing and adding new or update concepts and relations manually. In the code of server side, this functionality was developed with the help of a Java API, Jena Semantic Framework Ontology that supports OWL language, inspired by XML, in which the ontology was created.

In the Front-End application the technologies used were HTML, CSS to create all components that support the final product presentation. The interface is created in JSP to be able to integrate with the Java code. The final user client can be used in all modern browsers in the market, like Google Chrome, Mozilla Firefox or Internet Explorer.

## Requirements

As a proof of concept the system was developed based in requirements, which were divided in three types, Functional, Architectural, Technical (see Figure 5.2). The system developed was integrated in the Business and Construction domain, the main idea is to receive unstructured information and extract knowledge from it that could help to make autonomous enrichment of the ontology. Having this in mind, some of the functional requirements are frequent items-concept matching, that searches the ontology and tries to find keywords that are similar to the frequent items, consequently, if found retrieves the concept represented by this frequent item. To discover new concepts, this process is based in the previous and is executed when no concept is found related to the frequent item. In this case, the new concept found will be added to the Building & Construction domain ontology with the frequent items associated to it. The system must also be able to show the association rules with the important information in an easy way.



Figure . - System requirements



Figure 5.3 – Knowledge layer architecture

## Conceptual & Technical Architectures

The system developed to achieve the solution to the problem proposed is based on a Model View Controller (MVC) methodology. An MVC methodology is a method to present and organize the model in three components. In the Model component the proposed solution will hold all three databases that holds all the data for the system. In the first one, Document Repository, is the initial repository with all the initial unstructured data to be analysed. In the second database, namely Association Rules, is a database created to store the rules and all the analysed intermediate data like the concepts, and the metadata related with the associated rules, and all values related to the measures used to classify them. In the third one, specifically Ontology, is the main database to store the ontology itself. It will hold the concepts, and the relations to all of them.

View is the component that is responsible for the front-end. It is the system connection to the world. All the users will access the data by this interface.

Controller component is the core of the system. It represents the place where all the analysis is computed. In the present system proposal, one can see two cores. The first core is the Rapidminer Core, is responsible to make all the steps from the unstructured data until the association rules discovery process. The second Core is the Server where all the logic is developed. It is where all computation to deliver the requests from the users are made, linking all the information from the database to the users. It is the artificial intelligent component of the system.

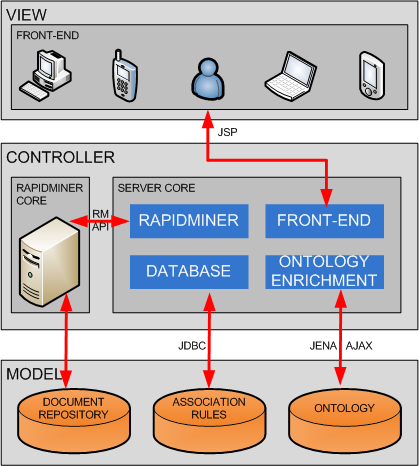


Figure 5.4 – System Architecture – MVC Methodology

## Implementation and Development

The present work proposes a solution for association rules discovery based in the following structure presented in . Each block represents a sub-process and was developed with a specific technology. ‘Document Analysis’, ‘FP-Growth’ and ‘Association Rules’ blocks were developed with Rapidminer processes. After this processing the results were delivered to the application by a rapidminer interface (Figure 5.5), coded in Ruby Programming Language technology.



Figure 5.5 – Rapidminer Main Process

The databases created were developed in MySQL and implemented in MySQL Workbench, a specific tool to develop databases. Both run on top of an Apache server.

The communication with the ontology is made with Jena Semantic Framework Ontology, a Java API that supports OWL language, in which the ontology is coded.

The Front End application is developed in NetBeans environment, and coded in Java Language with ASP features. Some of the technologies used can be found in Appendix X.

Protegè software was also used to aid the author to refer to the ontology structure.

ASSOCIATION

RULES

FREQ. ITEMSET MAPPING

ONTOLOGY

ASSOCIATION RULES DATABASE

DOCUMENT ANALYSIS

FP-GROWTH

Figure 5.6 – Conceptual Architecture

## Conceptual & Technical Architectures

The idea of this work is to implement a solution to help discover ontologic relations between concepts. Specifically, find Rules of Association for the concepts. The approach proposed lies on a conceptual architecture composed by several blocks, each one representing a process (see ). Namely ‘Document Analysis’, ‘FP-Growth’, ‘Association Rules’, ‘Frequent Itemset Mapping’ and the blocks representing two support databases ‘Ontology’ and ‘Association Rules Database’. In the following lines, each block will be explained in detail. This architecture is, in the author opinion, the one that best suits the problem.

### Document Analysis

Before one can discover the earlier ontologic relations, the source text documents with the unstructured information must be prepared in such way, so that can be understandable by this architecture. Some organization in them is necessary, along with some processing as the text is in a raw state. To achieve processed information, the documents go through some pre-processing in rapidminer (see ).



Figure 5.7 – Vector Creation Rapidminer Process

The first step of the preparation of the unstructured information is *Tokenize* process. The tokenize process is responsible for separation of the full text into a sequence of tokens. Tokens can be understood in several ways, for the purpose of this work, one can consider token as a set of letters. Everything that does not have letters, it is discarded.

The second step is the *Transform Case* process that has the main objective of transformation of all tokens to lower case. This is a necessary step so that all tokens that differ in a letter case would be considered the same and the respective frequency considered. For instance, “Token” and token” are considered the same, therefore will be transformed to “token”.

Next process is *Filter Stopwords* that is responsible of stopwords filtration. These stopwords are words that have no semantic importance for the context, like “*the*”, “*each*”, “*a*”, etc. All these stopwords are removed of the set of tokens.

The following process is the *Stem (Snowball)*, it is the stemming algorithm. This algorithm has the responsibility to transform the word in its stem, in other words, it will transform the word in its common morphological root. In this project the stemming algorithm used is the Snowball variation algorithm. This process can be optional, but one thinks it is of a great value, as it reduces the words to its stem, gathering them to the same word family to enrich its value in a document. Meaning that as more words are grouped for their stems, more representative is the stem in the document, consequently, reinforcing a better context of it. Another good advantage of the stemming process is the reduction of the data size augmenting each stem precision.

After all words are in their stem form, these set enters in the *Filter Tokens* process. This process makes a pruning operation in all tokens (words) that are lower than 4 and higher than 50 characters. This process is necessary to remove all unnecessary tokens that have no taxonomic relevance for the study, like chain of random letters, thus the author chose this interval as a fair number. Although one can consider relevant words with a minimum of 4 characters the upper value can be reduced to the maximum number of chars in each language. The value chosen of 50 chars as the biggest word is probably high for a word that also is stemmed, however this value was maintained for testing purpose. As an example, one can consider one of these tokens, for instance, the token “aaaaaaa”. It is bigger than 4 chars, and don’t have any relevant value for this project.

The last step of the document analysis is the generation of n-grams. The n-grams generation is the creation of sequences of 1 to N words, being for this case N=3, using unigrams, bigrams (eg. Waste Management) and trigrams (e.g. Electric Power Product). The purpose for this generation is a first try to find concepts and groups of words that represents concepts.

DOCUMENT ANALYSIS

Tokenize

Transform Cases

(lower cases)

Filter

Stopwords

Stemming

(Snowball)

Filter Tokens

(4-50)

Generate

n-grams (n≤3)

Figure 5.8 – Document Analysis Pipeline

The output of the analysis is saved into a temporary database for easement of processes. In the interface between the analysis and the FP-growth process, the results enters afterwards in a sub-auxiliary process Numerical to Binomial, whose function is to change the nominal values of the vector to binomial values, which changes to false every value inside an interval, and to true the ones outside. This means that words that have no significant ontologic meaning are filtered out of the document corpus.

Table 5‑1 – Numerical to Binomial regulation

|  |  |  |  |
| --- | --- | --- | --- |
| NumBinMax | Support | Confidence | Association Rules |
| 0,012 | 0,25 | 0,01 | 18 |
| 0,012 | 0,25 | 0,60 | 18 |
| 0,012 | 0,25 | 0,70 | 18 |
| 0,013 | 0,20 | 0,01 | 102 |
| 0,013 | 0,25 | 0,70 | 12 |
| 0,014 | 0,20 | 0,01 | 92 |
| 0,015 | 0,20 | 0,01 | 92 |

For the purpose of this work, the interval values were choose as follows: Min – 0.0 Max 0.013, as the shows, some tests were made to get the configuration of this module that gives a wider number of Association Rules to examine.

### FP-Growth

This block is responsible to find Frequent Patterns from the source documents that the author is analysing. The process uses the FP Growth algorithm, described earlier in this document. This process is the base for the recognition of frequent items that appear in the text more than others.

### Association Rules

(Mostrar alguns exemplos de aplicação das Association Rules no trabalho;)

Table 5‑2 – Association Rules Database Structure

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| # | Premise | Conclusion | Confidence | Conviction | Gain | Laplace | Lift. | Ps | Total Support |
| 1 | ConceptA | ConceptB | ValA | ValB | ValC | ValD | ValE | ValF | ValG |

### Frequent Itemset Mapping

(Explicar o que representa este bloco; Falar de onde é que aparece este procedimento, e baseado em quê; Explicar o que são os elementos que se apresentam em baixo)

The Frequent Itemset Mapping is a module that is executed after the rapidminer processing. The main objective is to map the concepts reaching of the association rules results with the concepts directly from the respectively ontology.

The mapping is processed by the similarity level calculus between frequent itemsets and ontology equivalent terms.

#### Frequent Itemsets

Frequent Itemsets are concepts that are found to be a frequent pattern from the source documents. This is a direct result from the FP-Growth algorithm.

#### Ontology equivalent terms

#### Similarity

#### Candidate concepts

(Concepts that are related to each other by the frequent items, in the ontology)

#### Mapping

(Transforming each frequent itemset in a concept that can be used in the ontology)

This process is responsible to map each frequent itemset from the source documents with the keywords that are associated with the concepts in the ontology.

### Ontology Enrichment

(Falar do processo de enriquecimento da ontologia, com a adição de novos conceitos, actualização dos já existentes; Como detectamos, o que actualizamos, etc)

(Falar o OWL como ferramenta de ajuda para visualização da ontologia)

## Front end

(Arranjar um nome para a aplicação)

(Falar da ligação entre Rapidminer->FrontEND<-JENA<-Owl(XML) )

(Inserir algumas imagens e explicar o funcionamento da aplicação através de diagramas, talvez Use Cases e/ou sequence diadrams[UML])



Figure 5.9 – Entity Relation Model

# Evaluation

(Falar da avaliação da aplicabilidade e importância do trabalho;)

## Use cases

## Scientific publications

(Falar da aprovação de publicações deste trabalho pela comunidade científica)

Incluir papers (e talvez incluir referência à ligação com o trabalho do Ruben e do Paulo)

# Conclusion and Future Work

# Bibliography

Agrawal, R., Imieliński, T., Swami, A., 1993. Mining Association Rules Between Sets of Items in Large Databases, in: Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, SIGMOD ’93. ACM, New York, NY, USA, pp. 207–216. doi:10.1145/170035.170072

Agrawal, R., Srikant, R., 1994. Fast algorithms for mining association rules, in: Proc. of 20th Intl. Conf. on VLDB. pp. 487–499.

Azevedo, P.J., Jorge, A.M., 2007. Comparing Rule Measures for Predictive Association Rules, in: Kok, J.N., Koronacki, J., Mantaras, R.L. de, Matwin, S., Mladenič, D., Skowron, A. (Eds.), Machine Learning: ECML 2007, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 510–517.

Azevedo, P.J., Silva, C.G., Rodrigues, J.R., Loureiro-Ferreira, N., Brito, R.M.M., 2005. Detection of Hydrophobic Clusters in Molecular Dynamics Protein Unfolding Simulations Using Association Rules, in: Oliveira, J.L., Maojo, V., Martín-Sánchez, F., Pereira, A.S. (Eds.), Biological and Medical Data Analysis, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 329–337.

Bayardo, R.J., Jr., Agrawal, R., 1999. Mining the Most Interesting Rules, in: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’99. ACM, New York, NY, USA, pp. 145–154. doi:10.1145/312129.312219

Bhujade, V., Janwe, N.J., 2011. Knowledge Discovery in Text Mining Technique Using Association Rules Extraction, in: 2011 International Conference on Computational Intelligence and Communication Networks (CICN). Presented at the 2011 International Conference on Computational Intelligence and Communication Networks (CICN), pp. 498–502. doi:10.1109/CICN.2011.104

Bonchi, F., Goethals, B., 2004. FP-Bonsai: The Art of Growing and Pruning Small FP-Trees, in: Dai, H., Srikant, R., Zhang, C. (Eds.), Advances in Knowledge Discovery and Data Mining, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 155–160.

Borgelt, C., 2005. An Implementation of the FP-growth Algorithm, in: Proceedings of the 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations. ACM, pp. 1–5.

Brin, S., Motwani, R., Ullman, J.D., Tsur, S., 1997. Dynamic Itemset Counting and Implication Rules for Market Basket Data, in: Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data, SIGMOD ’97. ACM, New York, NY, USA, pp. 255–264. doi:10.1145/253260.253325

Brin, S., Rastogi, R., Shim, K., 2003. Mining optimized gain rules for numeric attributes. Knowl. Data Eng. IEEE Trans. On 15, 324–338.

Costa, R., Figueiras, P., Paiva, L., Jardim-Gonçalves, R., Lima, C., 2012. Capturing Knowledge Representations Using Semantic Relationships An Ontology-based Approach. Presented at the SEMAPRO 2012, The Sixth International Conference on Advances in Semantic Processing, pp. 75–81.

Dhar, V., Tuzhulin, A., 1993. Abstract-driven pattern discovery in databases. IEEE Trans. Knowl. Data Eng. 5, 926–938. doi:10.1109/69.250075

Figueiras, P., Costa, R., Paiva, L., Jardim-Gonçalves, R., Lima, C., 2012. Information Retrieval in Collaborative Engineering Projects - A Vector Space Model Approach: Presented at the Knowledge Engineering and Ontology Development Conference 2012, SciTePress - Science and Technology Publications, Barcelona, Spain, pp. 233–238. doi:10.5220/0004139302330238

Fukuda, T., Morimoto, Y., Morishita, S., Tokuyama, T., 1996. Data mining using two-dimensional optimized association rules: Scheme, algorithms, and visualization. ACM SIGMOD Rec. 25, 13–23.

Gonçalves, E.C., 2005. Regras de associação e suas medidas de interesse objetivas e subjetivas. INFOCOMP J. Comput. Sci. 4, 26–35.

Google.com [WWW Document], 2013. URL https://www.google.com/ (accessed 7.7.14).

Guillaume Laforge, 2012. Groovy [WWW Document]. URL http://groovy.codehaus.org/

Han, J., Pei, J., Yin, Y., Mao, R., 2004. Mining frequent patterns without candidate generation: A frequent-pattern tree approach. Data Min. Knowl. Discov. 8, 53–87.

Hilderman, R., Hamilton, H., 2001. Knowledge discovery and measures of interest. Kluwer.

Hoque, A.M.S., Mondal, S.K., Zaman, T.M., Barman, P.C., Bhuiyan, M.., 2011. Implication of association rules employing FP-growth algorithm for knowledge discovery, in: 2011 14th International Conference on Computer and Information Technology (ICCIT). Presented at the 2011 14th International Conference on Computer and Information Technology (ICCIT), pp. 514–519. doi:10.1109/ICCITechn.2011.6164843

IBM - International Business Machines, 1996. IBM Intelligent Miner User’s Guide, Version 1 Release 1. SH12-6213-00 edition, July.

Koh, Y.S., Rountree, N., 2005. Finding Sporadic Rules Using Apriori-Inverse, in: Ho, T.B., Cheung, D., Liu, H. (Eds.), Advances in Knowledge Discovery and Data Mining, Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 97–106.

Korczak, J., Skrzypczak, P., 2012. FP-Growth in Discovery of Customer Patterns, in: Aberer, K., Damiani, E., Dillon, T. (Eds.), Data-Driven Process Discovery and Analysis, Lecture Notes in Business Information Processing. Springer Berlin Heidelberg, pp. 120–133.

Kumar, V., Chadha, A., 2012. Mining association rules in student’s assessment data. Int. J. Comput. Sci. Issues 9, 211–216.

Lavrač, N., Flach, P., Zupan, B., 1999. Rule evaluation measures: A unifying view. Springer.

Lin, D., 1998. An information-theoretic definition of similarity., in: ICML. pp. 296–304.

Lin, D., Pantel, P., 2001. DIRT@ SBT@ discovery of inference rules from text, in: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 323–328.

Liu, B., Hsu, W., Ma, Y., 1999. Mining Association Rules with Multiple Minimum Supports, in: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’99. ACM, New York, NY, USA, pp. 337–341. doi:10.1145/312129.312274

Mackie, J., 1977. Ethics: Inventing right and wrong. Penguin UK.

Manning, C.D., Raghavan, P., Schütze, H., 2008. Introduction to information retrieval. Cambridge university press Cambridge.

Marinica, C., Guillet, F., 2010. Knowledge-Based Interactive Postmining of Association Rules Using Ontologies. IEEE Trans. Knowl. Data Eng. 22, 784–797. doi:10.1109/TKDE.2010.29

Metanat Hooshsadat, SAMANEH BAYAT, PARISA NAEIMI, MAHDIEH S. MIRIAN, OSMAR R. ZA?ANE, 2012. UAPRIORI: AN ALGORITHM FOR FINDING SEQUENTIAL PATTERNS IN PROBABILISTIC DATA, in: Uncertainty Modeling in Knowledge Engineering and Decision Making, World Scientific Proceedings Series on Computer Engineering and Information Science. WORLD SCIENTIFIC, pp. 907–912.

Nakov, P., Hearst, M.A., 2008. Solving Relational Similarity Problems Using the Web as a Corpus., in: ACL. Citeseer, pp. 452–460.

Oxford University, 2006. Oxford Dictionary of English. Oxford University Press, London.

Paiva, L., Costa, R., Figueiras, P., Lima, C., 2013. Discovering Semantic Relations from Unstructured Data for Ontology Enrichment - Asssociation rules based approach. Presented at the CISTI’2013 - 8a Conferência Ibérica de Sistemas e Tecnologias de Informação, AISTI, Lisboa, pp. 579–584.

Pantel, P., Lin, D., 2002. Discovering word senses from text, in: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, pp. 613–619.

Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L., 1999. Discovering Frequent Closed Itemsets for Association Rules, in: Proceedings of the 7th International Conference on Database Theory, ICDT ’99. Springer-Verlag, London, UK, UK, pp. 398–416.

Piatetsky-Shapiro, G., 1991. Discovery, analysis and presentation of strong rules. Knowl. Discov. Databases 229–238.

Rapp, R., 2003. Word sense discovery based on sense descriptor dissimilarity, in: Proceedings of the Ninth Machine Translation Summit. pp. 315–322.

Salton, G., 1971. The SMART retrieval system—experiments in automatic document processing.

Salton, G., Wong, A., Yang, C.-S., 1975. A vector space model for automatic indexing. Commun. ACM 18, 613–620.

Silberschatz, A., Tuzhilin, A., 1995. On subjective measures of interestingness in knowledge discovery., in: KDD. pp. 275–281.

Spruit, M., 2007. Discovery of association rules between syntactic variables, in: Proceedings of the 17th Meeting of Computational Linguistics in the Netherlands. Citeseer.

Tan, P.-N., Kumar, V., Srivastava, J., 2002. Selecting the Right Interestingness Measure for Association Patterns, in: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD ’02. ACM, New York, NY, USA, pp. 32–41. doi:10.1145/775047.775053

Turney, P., Littman, M.L., Bigham, J., Shnayder, V., 2003. Combining independent modules to solve multiple-choice synonym and analogy problems.

Turney, P.D., 2006. Similarity of Semantic Relations. Comput. Linguist. 32, 379–416. doi:10.1162/coli.2006.32.3.379

Turney, P.D., Pantel, P., others, 2010. From frequency to meaning: Vector space models of semantics. J. Artif. Intell. Res. 37, 141–188.

Vo, B., Le, B., 2009. Mining traditional association rules using frequent itemsets lattice, in: International Conference on Computers Industrial Engineering, 2009. CIE 2009. Presented at the International Conference on Computers Industrial Engineering, 2009. CIE 2009, pp. 1401–1406. doi:10.1109/ICCIE.2009.5223866

W3C, 2004. OWL Web Ontology Language Overview [WWW Document]. OWL Web Ontol. Lang. URL http://www.w3.org/TR/2004/REC-owl-features-20040210/ (accessed 7.7.14).

Wang, K., Tang, L., Han, J., Liu, J., 2002. Top down FP-Growth for association rule mining. Springer.

Yao, Y., Chen, Y., Yang, X., 2006. A Measurement-Theoretic Foundation of Rule Interestingness Evaluation, in: Lin, P.T.Y., Ohsuga, P.S., Liau, D.C.-J., Hu, P.X. (Eds.), Foundations and Novel Approaches in Data Mining, Studies in Computational Intelligence. Springer Berlin Heidelberg, pp. 41–59.

Yih, W., 2009. Learning term-weighting functions for similarity measures, in: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2. Association for Computational Linguistics, pp. 793–802.

Zaki, M.J., 2000. Scalable algorithms for association mining. IEEE Trans. Knowl. Data Eng. 12, 372–390. doi:10.1109/69.846291

Zeng, B., Jiang, X.-L., Zhao, W., Luo, C., 2010. The improvement of weighted association rules arithmetic based on FP-tree, in: Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on. IEEE, pp. V4–549.

# Appendices

1. in literature, other names can be found to represent this same itemsets like antecedent and consequent (Hoque et al., 2011) in contrast to premise and conclusion. The latter designation will be adapted in the present study. [↑](#footnote-ref-1)
2. As a curiosity, Lift is well known in the scientific community as a measure used in the IBM’s Intelligent Miner (IBM - International Business Machines, 1996). [↑](#footnote-ref-2)
3. In the literature *PS* is also found under different designations, for instance, *Leverage* (Azevedo and Jorge 2007), *Rule Interest* (Gonçalves, 2005) or *novelty* (Lavrač et al., 1999). [↑](#footnote-ref-3)
4. Groovy builds upon the strengths of Java but has additional power features inspired by languages like Python, Ruby and Smalltalk. (Guillaume Laforge, 2012) [↑](#footnote-ref-4)