Delivering Scalable Frequent Pattern Mining for Non-Expert Data Miners

by

Zhao Han

A thesis submitted to the Faculty of Graduate Studies of
The University of Manitoba
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computer Science
The University of Manitoba
Winnipeg, Manitoba, Canada

August 2016

Thesis advisor

Dr. Carson K. Leung

Author

Zhao Han

Delivering Scalable Frequent Pattern Mining for Non-Expert Data Miners

Abstract

As a popular data mining task, frequent pattern mining has been proven to be helpful for non-experts. For example, mining frequent purchased products helps store managers increase sales. As another example, finding popular courses assists university administrators arrange courses to avoid schedule conflicts. However, many data mining researchers have focused on improving algorithmic efficiency, but have put less focus on providing non-experts with a system designed specifically for these non-experts. In my M.Sc. thesis, I propose such a system, called PatternShow, which consists of (i) a user-friendly frontend web interface along with a visualization tool called BundleVis to show effectively frequent patterns for non-expert miners and (ii) a cloud-enabled backend that offers scalable frequent pattern mining. Results of my user study show the effectiveness of PatternShow in delivering scalable frequent pattern mining for non-expert data miners.

Table of Contents

List of Figures

Acknowledgements

I would like to first express my love to my parents in China. Thanks for relentless support, financially and in spirit, throughout my B.C.Sc. (Hons.) and thesis-based M.Sc. studies. Without you, studying aboard in Canada and conducting my research would never be made possible. I am deeply indebted for your eternal love and respect.

For people in Canada, I would like to specially thank my academic supervisor, Dr. Carson K. Leung, and my girlfriend, Shihan Zheng. Dr. Leung introduced me to the exciting field of data mining. His enthusiasm in teaching and research, as well as his extensive knowledge, led me to graduate studies and kept inspiring me during my B.C.Sc. (Hons.) study and throughout my M.Sc. thesis research. His attentiveness to students and constructive feedback made my thesis progresses more smoothly. Shihan, along with her family in Canada, supports me in multiple ways. With her love, I continue to discover the surprising and wonderful moments of life.

In addition, I would thank my internal examiner, Dr. Yang Wang, and my external examiner, Dr. Qingjin Peng from Mechanical Engineering, for their constructive comments and suggestions towards this M.Sc. thesis. Also thanks Dr. Neil D.B. Bruce for chairing my M.Sc. thesis oral defense.

I also would like to say thank you to my lab mates in the Database and Data Mining Laboratory. Thanks Richard Kyle MacKinnon for proofreading and offering feedback while I fulfill my course requirements of my current M.Sc. degree. Thanks Hao (Thomas) Zhang and Fan (Terry) Jiang for making the life in the lab more fun. Thanks Fan again for sharing with me your positive previous experience on M.Sc. study and properly answering my questions.

For all others who come and go during my M.Sc. study, you are remembered.

Zhao Han

B.C.Sc. (Hons.), The University of Manitoba, Canada, 2014

The University of Manitoba
August 2016

—*for my parents*

Chapter 1

Introduction

Human beings are eager for knowledge because it empowers us and enriches our life. We used to gain knowledge through our experience and perception of the world, and in the last few thousand years, through education. In the current information age, computers allow us to collect and store a large amount of data. Due to their high volume, we may not be able to transform all those data into valuable information. Hence, data mining becomes a necessity to help us discover knowledge.

Data mining refers to the discovery of implicit, previously unknown and potentially useful knowledge [FPSM92]. *Frequent pattern mining* (*FPM*) [AIS93] is one of the fundamental tasks during the data mining process and finds one kind of knowledge called frequently occurring *sets of items* (shorthanded as *itemsets*). For example, a store manager may want to find what products customers frequently bought with diaper from the store transactional database. Assuming beer is one of the frequently bought products with diapers, the store manager can put several boxes of beer closer to a bag of diapers to increase profit. University administrators may want to find popular courses frequently taken together by students, which helps course scheduling and avoid course time conflicts [LJI11]. Indeed, frequent pattern mining is interesting because it is applicable as long as an item has the intrinsic property—namely, frequency. Unfortunately, there are currently no systems readily available for store managers and those *non-expert miners* who do not have sufficient computer science background to mine and better interpret the meaning of frequent patterns.

The advancement of frequent pattern mining has been focused on algorithm studies. Those algorithms are either improving previous work [PCY95, LLN00] or novel in terms of data structure [HPY00, PHL+01]. Newer algorithms are often more efficient in terms of memory or disk space, CPU cycles, or data characteristics (e.g., dense vs. sparse). However, these algorithms are mostly developer-oriented or researcher-oriented. They are not immediately available, leaving readership to figure out implementation details. Furthermore, the algorithms usually return a long textual list of frequent patterns, which is difficult to understand.

In the last decade, several visualization techniques [LJI11, LIC08b, LJ12, BSH13, PW14] for frequent patterns have been proposed to help users better interpret the long textual list of frequent patterns. However, those visualization techniques are not available to public and leave users to implement on their own. After implementation, the software is required to be installed and run on users' machines, and hence relies on the limited resource available. As the data volume continues to grow, users have to invest in their own infrastructure to get and interpret mining results. At the same time, individuals and small and medium business owners cannot afford and do not have the expertise to plan and implement the visualization techniques and the infrastructure.

Cloud computing has been the new wave of computing paradigm and became mature in the past decade. In this M.Sc. thesis, I deal with the aforementioned issues by using cloud computing.

Cloud computing offers dynamic scalability through virtualization, which is capable and favorable of handling a large quantity of transactional data ranging from gigabytes to terabytes. Its pricing model, which derives from utility computing, allows users to pay on a per-use basis and thus helps avoid upfront fees for computing resources. Compared to the visualization systems, users are not required to install on their own machines in order to use the software, and hence do not have to rely on the limited resource available on users' machines. To get mining results faster, users can leverage the infrastructure on the cloud and do not have to plan and invest in their own infrastructure, which they may not have the expertise to do so.

Due to the prevalence of Internet used in cloud computing, a cloud-enabled web interface allows the service accessible from everywhere, every type of device, and all kinds of users with different expertise. This is important because the majority of users tend to use more than one device (e.g., a laptop and a smartphone such as Android or iPhone). Different levels of users are used to web interface, such an interface makes the system easy to use from the start. For the practitioners developing the service, they can also leverage existing cloud solutions such as Amazon Web Service to ease the development effort. Last but not least, the interface also hides the complexity of frequent pattern mining algorithms themselves and removes the burden of choosing an algorithm most suitable for the data, which might be dense or sparse, at hand, making users feel like home.

The graphical web interface offers *abstraction*, which helps users better understand mined frequent patterns and hides the underlying complexity. The cloud-based backend provides *scalability*, which helps practitioners ease the development of such an interface and, more importantly, shortens the execution time to mine frequent patterns. Hence, I would like to explore the following research questions in this thesis:

1. Can we make a data mining system readily available for non-expert miners to mine frequent patterns?

2. Can we make a visualization tool readily available for non-expert miners to better interpret mined frequent patterns?

3. Can we do both in a system that is scalable and non-expert miners do not have to worry about the computing infrastructure?

4. Can we make the system user-friendly in terms of effectiveness and efficiency?

Before answering those questions, we need to understand who are non-expert miners. *Non-expert miners* are those who do not have sufficient computer science knowledge to implement a frequent pattern mining system to mine frequent patterns and to develop a frequent pattern visualization tool to better understand frequent patterns. However, frequent pattern mining do help them, for example, schedule courses or know what are the popular products their customers usually buy together. In other words, frequent pattern mining benefits non-experts. Saying this does not necessarily mean that there is no need from experts but means that experts have the ability to implement themselves to meet their need. Hence, non-expert miners are the main users I want to satisfy in this thesis.

1.1 Thesis Statement

To answer the above research questions, I design, implement, and evaluate such a cloud-enabled web application called PatternShow—along with a novel frequent pattern visualization called BundleVis—that is scalable and readily available for non-expert miners to mine and easily interpret frequent patterns in this M.Sc. thesis.

1.2 My Contributions

The following are three key contributions in my thesis:

1. My proposed design and implementation of a cloud-enabled web application is, to the best of my knowledge, the first frequent mining system that is designed for and readily available to non-expert miners so that they can mine for frequent patterns without worrying users' own computing infrastructure.

2. The novel frequent pattern visualization tool that is integrated into the system is also the first that specifically targets to non-expert miners to help them better comprehend frequent patterns, and is readily available to use.

3. The evaluation I conducted is, to the best of my knowledge, the first detailed user study that reveals the effectiveness and efficiency of a non-expert frequent mining system and non-experts' thoughts.

1.3 Thesis Outline

This thesis consists of five chapters. Chapter 2 (Background) and Chapter 3 (Related Work) establish the context of my thesis. Chapter 4 and Chapter 5 explain the visualization part and the architecture of PatternShow. Chapter 6 describes the user study that I conducted in order to evaluate the effectiveness and efficiency of PatternShow. The final chapter draws a conclusion and some future research directions.

To ease reading and understanding my contribution, Chapter 2 covers fundamental concepts and terminology related to this thesis in the hope of clarifying assumed knowledge. To unify the understanding of those concepts between me and the readership, I give definitions of data mining, frequent pattern mining and cloud computing, and the common terminology used in each. To have a concrete understanding, examples are drawn and applications to various domains are given.

In Chapter 3, I discuss related work in the literature. I categorize them to four broad categories. I start by discussing researchers' early attempts to deliver data mining models to a wider audience through Internet and, thus, to lower barriers to data mining tasks. I then do an extensive review of all existing frequent pattern visualizations that are the important parts for non-expert to comprehend frequent patterns. Afterwards, I summarize researchers' effort into proposing software that specifically targets layman data miners. I finish this chapter by surveying the work done by researchers who apply frequent pattern mining to specific fields. The applications serve as motivation and examples to show users what can be done.

With all the background and the related work discussed, subsequent chapters cover all my original work aimed to be readily available to non-experts and user-friendly. Chapter 4 introduces BundleVis, a novel visualization for frequent patterns that runs in browsers. Accompanied by screenshots, basic visualization elements, and a simple yet complete example are explained to get familiar with all the features available in the visualization. I then compare BundleVis with existing frequent patterns visualizations and describe several main advantages of BundleVis over the existing ones. Afterwards, I conduct case studies on real datasets to draw real-world examples on how BundleVis can be used to answer questions. The course dataset used in the case study for BundleVis is appended.

In Chapter 5, I discuss PatternShow, a novel and scalable application to mine frequent patterns. I discuss the architecture (which makes the whole system scalable) and the web interface (which is essential to non-expert users). To explain the architecture, a diagram of the workflow is given and described in the order of overview and detailed subcomponent. For the web interface, it mainly provides user data management. As expected by users, it allows users to register, login, remember password, retrieve password, upload data, and access data. BundleVis is also integrated into the web interface.

Before concluding my thesis, I describe the procedure to evaluate PatternShow and analyzed the results in Chapter 6. The dataset is also described. The material presented to participants are appended in Appendix B. In the analysis, the data collected from each task and each question from the user satisfaction questionnaire are visually presented to users using bar charts.

Finally, I conclude my thesis by answering each of the research questions I set earlier and discuss future work in Chapter 7.

Chapter 2

Background

In this chapter, I explain the terminology and several important concepts used throughout my thesis to help readers to understand the contribution of my thesis. Definitions and examples of data mining, frequent pattern mining, and cloud computing are given.

2.1 Data Mining

Data mining refers to the discovery of implicit, previously unknown and potentially useful knowledge [FPSM92]. As mentioned earlier, *frequent pattern mining* (*FPM*) [AIS93] is one of the fundamental tasks in the field of data mining. Data mining includes others tasks such as clustering, classification, and outlier detection.

Clustering is to answer how to put similar data into groups or clusters. To do that, there needs to be some similarity or dissimilarity measure. In a practical manner, advertisement providers may want to deliver some commercials only to a targeted audience to make the advertisements effective. Given a customer database with properties of those customers (e.g., age, purchase history), clustering algorithms can automatically separate customers to several similar groups, where advertisement providers can choose the most effective group as the audience for the specific advertisement.

Classification, on the other hand, is to categorize data into existing groups. One famous example is that, after we know an existing group of cat pictures, how can we recognize another cat? This example is applicable to face recognition. For the terminology, the existing groups are called training data; those cats that we never see before are called test data. The actual algorithm implementation is called a classifier. Given that a user put spam emails to a folder, a spam classifier can then put suspicious incoming emails into that folder based on its content and the properties of those emails (e.g., email host).

It is worth noting that clustering and frequent pattern mining can be considered as unsupervised learning tasks. Learning here means finding patterns in data (e.g., environment). They are unsupervised because no prior data given before mining and data are mined with only its intrinsic properties (e.g., distance and frequency). In contrast, classification is one type of supervised learning, because training data is explicitly given during the learning process.

2.2 Frequent Pattern Mining

The resulting data mined for by frequent pattern mining algorithms are frequent sets of items (i.e., itemsets), which are also known as frequent patterns. To elaborate, frequent itemsets refer to frequently occurring sets of items, and are usually mined from a transactional database.

A transactional database consists many transactions. One transaction is the products purchased during one visit to a store or supermarket such as Walmart. If a customer purchased apple, pear, lettuce, beer, milk and diaper, the transaction is denoted as {apple, pear, lettuce, beer, milk, diaper}. Note that even though customers may choose products by a specific order, when a transaction is recorded to database by a cashier at the counter, the products are not ordered. In addition, the quantity information is also eliminated during the mining process in order to simplify the mining process.

To distinguish from frequent items from non-frequent items, a frequency threshold is specified, called *minimum support*. Minimum support is often shortened to *minsup*. In this example database, a frequent pattern mined would be {beer, milk, diaper}, if the number of customers who buy those items together equals or exceeds *minsup*.

The number of unique domain items appearing in all frequent itemsets is usually indicated by m, and the notation of k-itemsets is often used to indicate a collection of itemsets with the same cardinality (i.e., the size of an itemset).

On the other hand, frequent itemset visualizations are graphical representations of frequent itemsets. Visual representations can be compared to textual representations that are long lists of itemsets generated by frequent pattern mining algorithms (e.g., Apriori [AS94], FP-tree [HPY00]).

2.3 Cloud Computing

Cloud computing has been a popular term these days both in the industry and among researchers in Computer Science. Even though the literature on cloud computing has been active and attempted to define the term, there is no consensus on this yet. However, being part of the background knowledge, I adopt one of the common definitions given by Foster et al. [FZRL08] based on paper citation count.

As Foster et al. [FZRL08] stated, cloud computing is essentially a distributed computing paradigm, driven by economies of scale, which has a large-scale abstracted IT resources delivered on demand to customers over the internet. There are some key concepts in this definition. Distributed computing is to run programs on multiple computers (i.e., computer cluster), but to users, the cluster appears as a single computer [TVS01]. Economy of scale means that, as the customer base and the scale of a service increase, the cost paid by each customer decreases. IT resources aforementioned can be computing power, storage, platforms (e.g., Google App Engine for running Java web applications) and services (e.g., Google Docs). IT resources are abstracted in the way that the physical compute recourses (e.g., CPU power, memory, and disk) are virtualized. Dynamic scalability comes from that virtualization on physical IT resources allows the dynamic provision of resources based on CPU load or memory/storage usage based on users.

Due to the abstraction through virtualization of physical IT resources, the service delivered to customers can be classified as three layers: *Infrastructure as a Service* (*IaaS*), *Platform as a Service* (*PaaS*) *and Software as a Service* (*SaaS*) [SSW10]. IaaS offers IT resources. Amazons' Elastic Compute Cloud (EC2) and Simple Storage

Service (S3), used in my thesis work, are two examples. EC2 provides processing power and S3 provides storage resources. PaaS is another abstraction layer sitting between IaaS and SaaS, providing platforms for applications. For example, Heroku[1] offers managed runtime (e.g., Ruby on Rails[2], Java Play Framework[3]) management for applications to ease deployment and scalability effort. SaaS is the most visible layer and directly faces end-users, not developers. Google Docs[4] and Microsoft's outlook.com[5] are two examples.

Because my thesis work is for non-expert data miners, PatternShow is categorized into SaaS and leverages IaaS providers: EC2, S3 and Amazon Elastic MapReduce (Amazon EMR)[6]. Their use is described in Section 5.1.

2.4 Summary

In order to understand my work PatternShow, which can be considered as a frequent pattern mining application, we must have some background knowledge on frequent pattern mining. Hence, in this chapter, I explain what data mining is. Similarly, I also explain what cloud computing is because it is needed to allow PatternShow to scale for a large amount of data.

More specifically, in this chapter, I reviewed some frequent pattern mining concepts. Frequent mining algorithms usually mine from a transactional database consisting of many transactions, and resulted in frequently occurring sets of items, i.e.,

[1] https://www.heroku.com/

[2] http://rubyonrails.org/

[3] https://www.playframework.com/

[4] https://docs.google.com

[5] https://www.outlook.com

[6] https://aws.amazon.com/elasticmapreduce/

frequent itemsets. The frequentness threshold used is called minimum support. Frequent mining algorithms usually give long lists of itemsets, while frequent itemset visualizations are graphical representations of frequent itemsets.

In addition, I also reviewed the concept of cloud computing, which delivers abstracted IT resources on demand to customers over the internet. Cloud computing is used because it allows processing a large amount of data. Three layers are defined: IaaS (Infrastructure as a Service) offers IT resources, PaaS (Platform as a Service) provides platforms for applications, and SaaS (Software as a Service) directly faces end-users, not developers. Because my thesis work is for non-expert data miners, PatternShow is categorized into SaaS and leverages IaaS providers.

Chapter 3

Related Work

In this chapter, I discuss related work on delivering data mining models over the Internet, frequent pattern visualizations, providing data mining to non-experts, and frequent pattern mining applications. Delivering data mining models over the Internet is necessary to make data mining accessible to a wider audience by leveraging the popular web. Frequent pattern visualizations are needed for non-expert users to better interpret mined result, frequent patterns. My work aims to provide data mining to non-experts makes us focus on meeting non-expert miners' need. At last but not least, various frequent pattern mining applications are surveyed as they are the motivation that drives us to combine and continue the aforementioned work to deliver frequent mining to non-expert miners.

3.1 Delivering Data Mining over Internet

Delivering data mining as a service, an attempt to lower barriers to data mining tasks, is not a new concept. Some researchers attempted to push data mining models to a wider audience through the Internet.

Sarawagi and Nagaralu [SN00] first proposed providing data mining models as a service over the Internet. Data mining models are usually extracted from a large amount of data and typically limited to those who have enough data and expertise. Even though the Internet makes information sharing thrive, various types of data and models are still hidden in disconnected databases. Only a few of them are embedded in software and the only way to use those models is to purchase.

To make data mining models more accessible, Sarawagi and Nagaralu [SN00] proposed the Internet service approach which encourages further sharing and improves accessibility. Compared to data mining models embedded in purchased software, the models provided via Internet can easily be up to date, leading to higher accurate results. Other advantages include installation-free and low cost for occasional use. To make the data mining services useful, Sarawagi and Nagaralu analyzed several realistic example scenarios and summarized three challenges to be solved before the model services become a reality. The first is the standardization of data and model, including easily interpretable input and output format of data and model structure. This makes deploying combined collaborative models on the Internet easier. Secondly, sharing enterprise and user data or models should be confidential. Summarized models instead of raw data are preferred. The last two challenges are around integrating models. One is to integrate those of the same domain but provided by different parties. The other is to integrate existing models to a personalized model from an end-user.

In the aforementioned example scenarios, the concentration is on the applications of the model services (e.g., recommendation services and risk prediction service). However, as FPM is a fundamental task during the data mining process, it is a model by itself and the model has wider applications. Hence, data mining model standardization, sharing and integration will not be the focus of my thesis work. In terms of similarity, my work is also based on Internet and hence shares some benefits such as improved accessibility, installation-free and low cost for occasional use.

Xu and Zhang [XZ05] abbreviated data mining model service to *knowledge-as-a-service* (*KaaS*). In general, a knowledge consumer can query a knowledge server that then send back an answer supplied by underlying knowledge models.

In later years, researchers proceed within the knowledge service paradigm. For instance, Gorea [Gor08] focused on enabling reuse and sharing of data mining models and proposed an online knowledge provider. Lai et al. [LTC12] presented a business case study that a private knowledge network is deployed as KaaS within the medical industry. A pay-as-you-use private knowledge system is developed to offer knowledge from consultants to hospitals and doctors. However, Lai et al. [LTC12] considers knowledge in KaaS not provided by underlying data mining models but summarized or directly from human experts, which makes it off the track of the original purpose of data mining model services.

Amaro et al. [ALSG12] explored KaaS as a way to communicate between knowledge server and smart TVs to provide personalized recommendation and user directed advertising services. The underlying model on the knowledge server is constantly improving based on users' activity data and need to be delivered to TV users. To facilitate communication between TV and the knowledge server, the paper focuses on the sematic description for data and models using OML (Ontology Markup Language) to specify them. However, the knowledge server is private to TV suppliers and thus not shared, which is against the original intent of knowledge (i.e., data mining models) sharing and will not push data mining models to a wider audience.

Grolinger et al. [GMCE15] presented a case study of KaaS in the disaster management domain. Previously, disaster data management is not scalable to large disaster-related data and heterogeneous sources. To efficiently store (i.e. high availability) and manage data for data integration, analysis, and decision-making, NoSQL database technologies and integration of multiple cloud providers were explored. However, the case study focused on database scalability and data integration instead of data mining models extracted from datasets.

This line of research focuses on sharing and delivering data models to a wider audience. However, majority of papers on KaaS applications attempted to make domain specific work done, limiting themselves to a narrow audience, and thus against the original intention of Sarawagi and Nagaralu: sharing data mining models. In addition, researchers did not provide graphical user interface (GUI), which is important to reach non-expert miners.

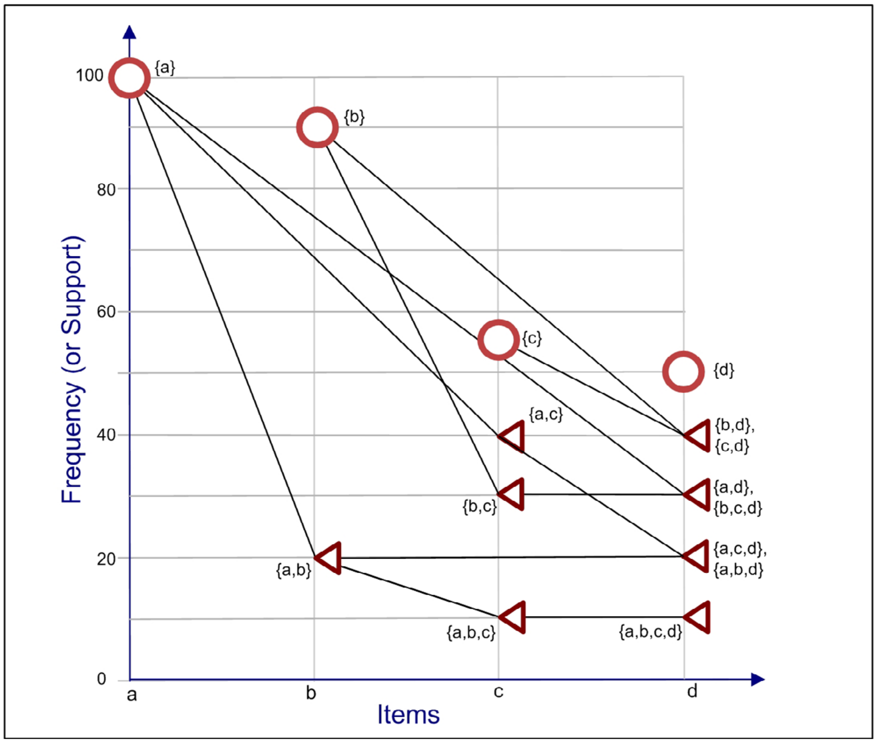


Figure 3.1: FIsViz [LIC08a]

3.2 Frequent Pattern Visualizations

Leung et al. [LIC08a] proposed *FIsViz* (Frequent Itemset Visualizer) to present frequent itemsets in graphical format. As shown in Figure 3.1 [LIC08a], frequent itemsets are represented as polylines with domain items on the x-axis and item frequencies on y-axis. By doing this, users can easily see the frequent itemsets of each domain item. By connecting itemsets with their supersets, users can find their related frequent itemsets, i.e., frequent supersets and subsets of an itemset. Because some polylines are overlapped, users can also see the containment relationship between them (e.g., $\{a, b\}$ and $\{a, b, c, d\}$). However, overlapped polylines sometimes represent different frequent patterns (e.g., $\{a, c, d\}$ and $\{b, c, e\}$ vs. $\{a, c, e\}$ and $\{b,$

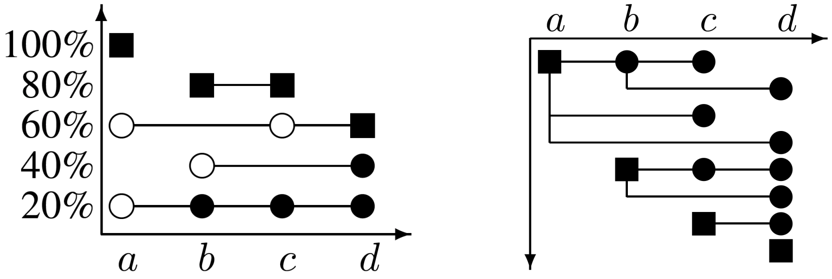


Figure 3.2: WiFIsViz. Left is overview. Right is detailed view. [LIC08b]

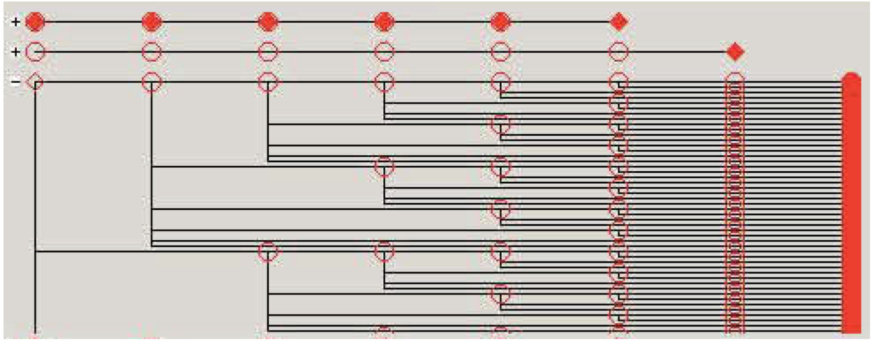


Figure 3.3: FpViz [LC09]

c, *d*}).

Leung et al. [LIC08b] also proposed *WiFIsViz* (Wiring Frequent Itemset Visualizer). As seen in Figure 3.2 [LIC08b], a frequent itemset is represented by a wire, which is a horizontal line of connected circles where each circle indicates one item in the itemset. To avoid intersected lines, certain horizontal lines get merged when the lines share a common prefix or the represented itemsets have the same frequency. Because the compression technique is lossy, a detailed view is provided on the half screen.

Afterwards, Leung and Carmichael proposed *FpVAT* [LC10] and *FpViz* [LC09] to further improve WiFIsViz. In FpVAT, the horizontal lines are also used for representing raw transaction data. FpViz is developed to represent a wide range of frequent patterns (i.e., frequent patterns with a large number of distinct frequencies). Previously, lines representing them can be crowded and thus go off screen in y-direction. To reduce the number of horizontal lines, FpViz merges those frequent itemsets with the same frequency and the same prefix, leaving lines with multiple solid circles indicating multiple itemsets on a single line while the line width shows how many lines are merged. Unlike WiFIsViz using up half-screen space to show details of all merged lines, users can interactively expand a single horizontal line to show all lines merged into that line. By doing so, users can see more lines horizontally, making the visualization more scalable.

In addition to line-based visualizations, researchers also explored treemap [JS91] and circular layouts. For example, Leung et al. [LJI11] proposed *FpMapViz* to make the hierarchy (i.e. prefix and extension relationship) of frequent patterns more explicit. Unlike FpMapViz, horizontal line representations (e.g., FIsViz and FpViz) lose the linkage of the prefixes and extensions of frequent patterns because users have to expand the horizontal lines to see the relationships between merged frequent patterns. To show the hierarchy of frequent patterns clearly, the tree map technique is applied. A tree map is the representation of a two-dimensional tree where child nodes are displayed inside their parent node and the size of a child node is proportional to its siblings and parent. As shown in Figure 3.4, FpMapViz represents a frequent pattern in a square and recursively shows all itemsets inside its superset (e.g., $\{a, b, c\}$ inside $\{a, b\}$); different colors are used to indicate frequency. To make FpMapViz

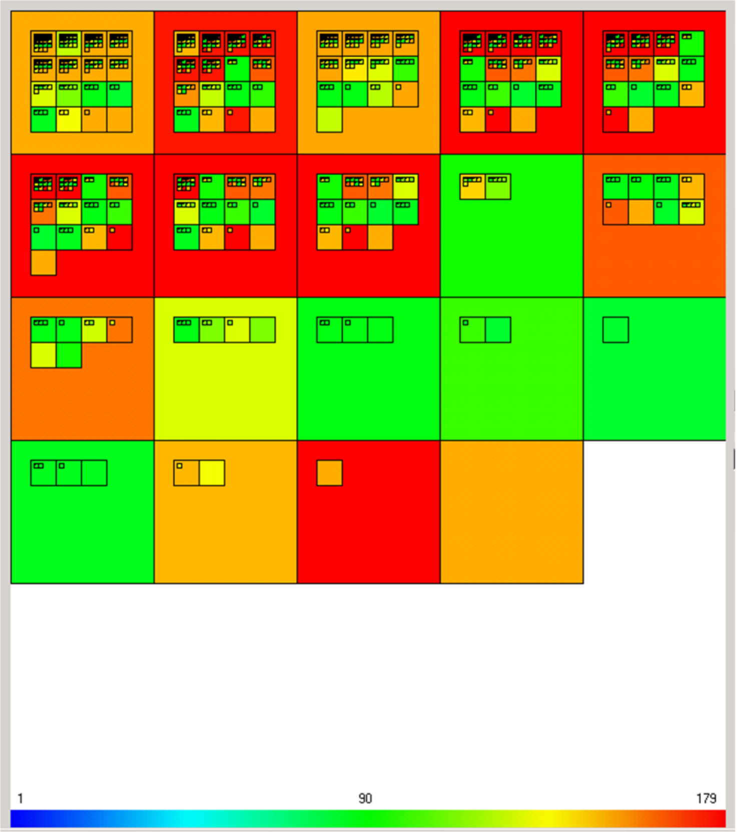


Figure 3.4: FpMapViz [LJI11]

scalable, users are allowed to zoom in or out a square to inspect different *k*-itemsets.

Leung and Jiang [LJ12] later proposed *RadialViz* to assist comprehend frequent patterns at different viewing orientations. Previously, people sitting around a table will not understand the same visualization, for example, shown on a tablet. A business analyst may see the visualization upside down, while a store manager may see the visualization right-side up. As shown in Figure 3.5, RadialViz has an orientation-free radial layout where pie segments represent frequent patterns. While the radius shows the frequency, color indicates the number of items. By doing so, users at different orientations can distinguish patterns with different frequencies through high and low

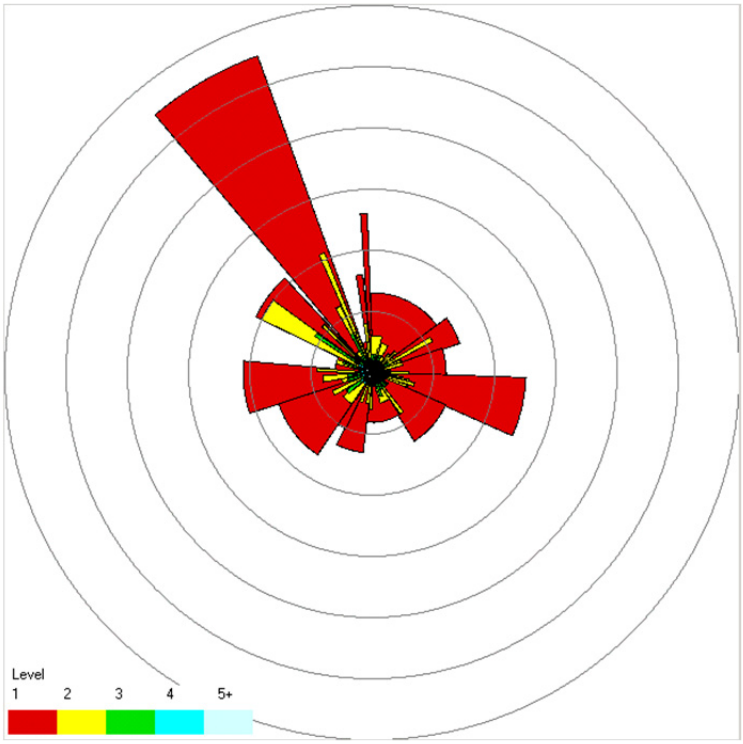


Figure 3.5: RadialViz [LJ12]

pie segments and interpret the cardinality information through segments in deeper or lighter colors. Similar to tree map, $(k-1)$-itemsets are placed inside k-itemsets to show the prefix/extension information. In addition, users can zoom in a small pie segment or zoom out to see different k-itemsets.

Bothorel et al. [BSH13] introduced *multi-circular graph* to visualize large quantity frequent patterns. They avoid representing a frequent pattern on a potentially very long line. Shown in Figure 3.6, frequent patterns are represented by nodes on nested circles which represent k-itemsets. Circles are nested where k-itemsets are placed inside $(k-1)$-itemsets. Nodes on two adjacent circles are linked if inner nodes are subsets of the outer node. Note that the links between circles are bundled to make

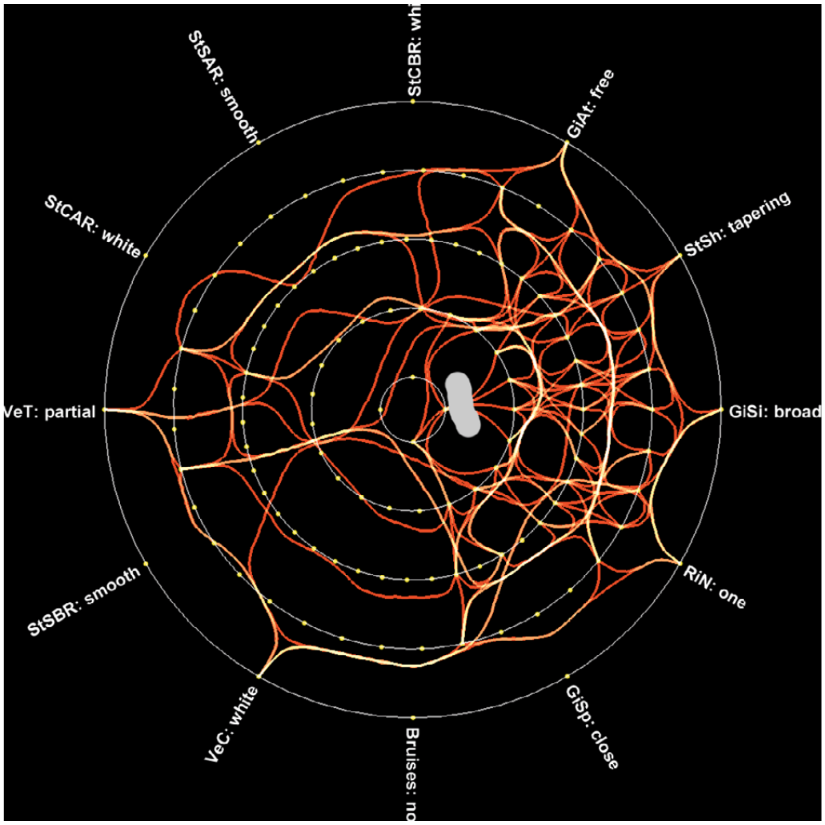


Figure 3.6: Circular graph [BSH13] (Used with permission of Springer)

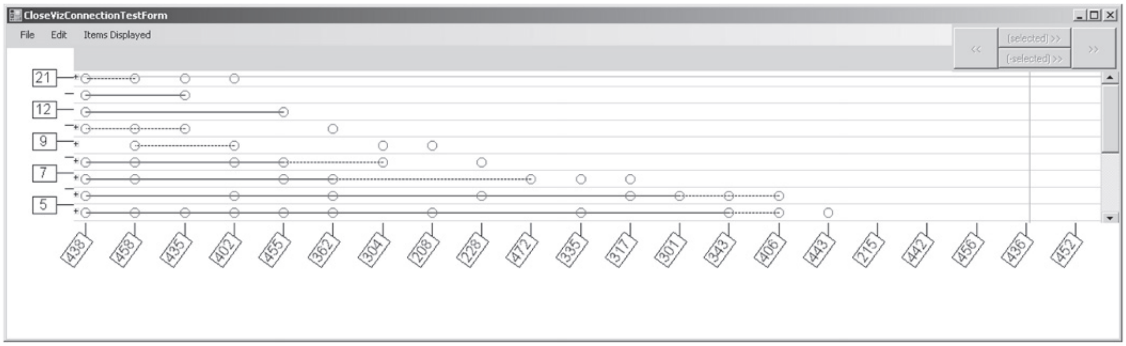


Figure 3.7: CloseViz [CL10]

the graph simpler and easier to comprehend. Moreover, the color of a node is used to indicate the frequency of the frequent pattern, and users can select a node to highlight

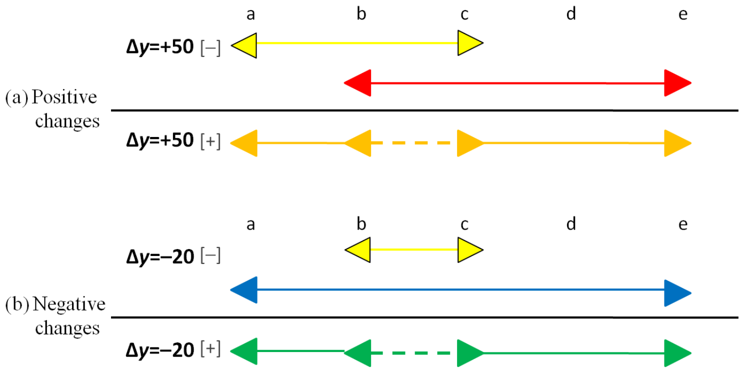
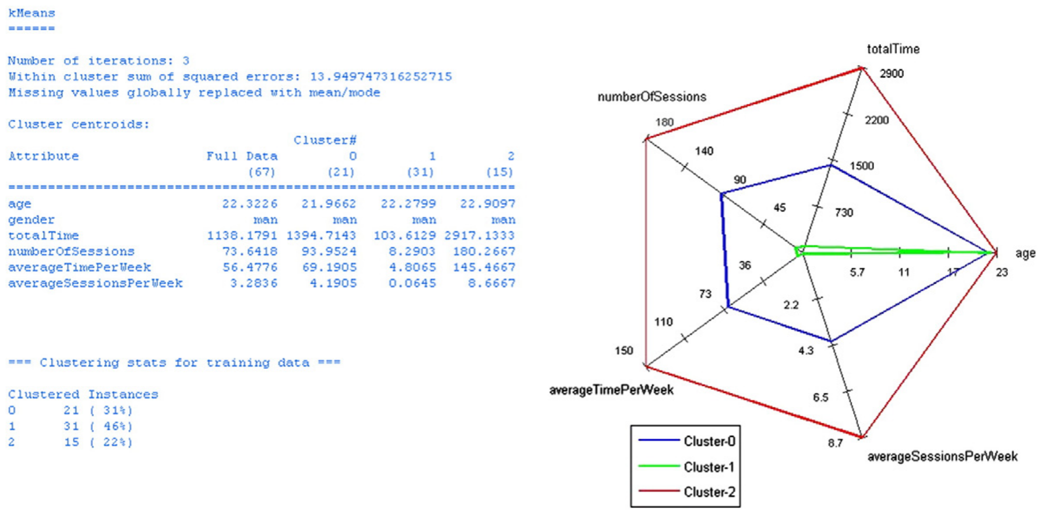


Figure 3.8: ContrastViz [CHL11]

its links.

Other visualizations, CloseViz and ContrastViz try to solve thing specifically. In Figure 3.7, CloseViz [CL10], a line-based visualization similar to FpViz, attempted to reduce the number of frequent pattern representations by mining and showing only closed frequent patterns (which do not have supersets with the same frequency). ContrastViz [CHL11], as shown in Figure 3.8, attempted to visually compare different set of frequent patterns.

As discussed earlier, those visualizations are not available to non-experts and require developer knowledge to implement. In my thesis work, I included an interactive frequent pattern visualization tool, BundleVis, with which non-experts can use immediately. I discuss BundleVis and compare it with those existing visualizations.



```
kMeans
======

Number of iterations: 3
Within cluster sum of squared errors: 13.949747316252715
Missing values globally replaced with mean/mode

Cluster centroids:
                                  Cluster#
Attribute               Full Data         0         1         2
                             (67)      (21)      (31)      (15)
================================================================
age                       22.3226   21.9662   22.2799   22.9097
gender                        man       man       man       man
totalTime               1138.1791 1394.7143  103.6129 2917.1333
numberOfSessions          73.6418   93.9524    8.2903  180.2667
averageTimePerWeek        56.4776   69.1905    4.8065  145.4667
averageSessionsPerWeek     3.2836    4.1905    0.0645    8.6667



=== Clustering stats for training data ===

Clustered Instances
0       21 ( 31%)
1       31 ( 46%)
2       15 ( 22%)
```

Figure 3.9: A glimpse of the software developed by Zorrilla et al. [ZGS13] (Used with permission of Elsevier)

3.3 Providing Data Mining for Non-experts

Zorrilla et al. [ZGS13] were the first researchers to introduce data mining service specifically targeted to non-expert miners. Previously, data mining projects were costly to companies and need expert data miners to implement. Zorrilla et al. proposed a service oriented architecture and developed a web interface for online education, as shown in Figure 3.9. Because of the nature of service orientated applications, they separate the application to multiple layers that are responsible for integrating various data sources, exposing interfaces to underlying functionalities such as data preprocessing, data mining algorithms and data visualization, as well as GUI. For each functional feature (e.g., grouping students based on web logs and frequently accessed resources), an underlying template is provided with input parameters that are selected after intense and extensive experimentation on data mining algorithms.

The web interface includes several types of charts such as histograms, spider and pie charts. Even though the web interface is user-friendly and the service-oriented architecture makes the application easily extendable, each template needs a considerable amount of work from expert miner. In addition, as stated in the paper [ZGS13], the system is not designed for dealing with the large quantity of data and, thus, not scalable in the big data era. Zorrilla et al. later improved the application [GSPZ14] but only in terms of functionality.

Ramya et al. [RS14] developed a knowledge extraction system for non-expert miners, but the system only generates a trivial report with simple line graphs, while data mining puts emphasis on extraction of implicit and previously unknown information. Guedes et al. [GMF06] also developed a service-oriented architecture for data mining, offering a simple interface to users and supporting computationally intensive processing through parallelism. Experiments show that (i) Anteater is 16 times faster than non-distributed systems and (ii) novice users understand the mined results without further data mining knowledge.

The algorithm included in my thesis work is also distributed. Meanwhile, the system is specifically designed for frequent pattern mining. One reason is that it has wide applications, which are surveyed in the next section.

3.4 FPM Applications

Because frequent pattern mining (FPM) is a generic field and frequent patterns can be found in a variety of data, the applications of frequent pattern mining are of broad range and keep springing up. This also leads to the potentially wide use of my proposed application. In 1994 [AS94], the original frequent pattern mining paper has the first application: analyzing customer buying behavior, which was described in Chapter 1. Chen et al. [CPY98] first applied frequent pattern mining to weblogs: frequent access patterns found in log data can help maximize visitors' accesses [CPY98] and differentiate students' access behaviors for online learning [ZL01]. The access patterns can also be used for personalization which recommends web pages [MCS00]. Frequent pattern mining can also be applied to time series data. Han et al. [HDY99] attempted to find periodic patterns in temporal data, while Bettini et al. [BWJL98] use FPM to detect frequent event patterns. Data mining researchers also applied FPM for clustering. Wang et al. [WXL99] proposed a clustering algorithm for transactional database by finding frequent overlapping partial transactions. These applications of frequent pattern mining tend to be domain specific and the resulted achievements are not immediately available to non-expert miners.

3.5 Summary

In this chapter, I organized related work into four broad categories: delivering data mining over Internet, frequent pattern visualizations, providing data mining for non-experts, and FPM applications.

Sarawagi and Nagaralu first proposed to provide data mining models over Internet to lower entry barriers and make data mining more accessible to non-expert users. However, Sarawagi and Nagaralu's blueprint is broad and the fundamental data mining task, FPM, is missing.

Except the convenient delivering over Internet, visualization rather than textual results is preferable. I surveyed many frequent pattern visualizations, including line-based visualizations (FIsViz, WiFIsViz, FpVAT and FpViz), space-filing visualizations (FpMapViz), circle-based (RadialViz), graph-based visualization by Bothorel et al., and some other specialized visualizations for closed frequent patterns and comparing frequent patterns. Most of them fall short because they are not readily available to non-experts and require developer knowledge to implement.

In the last three years, researchers start developing integrated systems to provide data mining to non-experts. Zorrilla et al. developed both the user interface part and the underlying mining system. The user interface part includes several types of charts such as histograms, spider and pie charts. The mining system consists of a few data mining model templates heavily developed offline by experts. This leads to the difficulty to extend.

Chapter 4

BundleVis: Novel Visualization

Frequent pattern visualization is an integral part of any user-friendly frequent pattern mining system. In this chapter, I present *BundleVis*, a novel visualization tool for frequent patterns.

BundleVis is a visualization tool written in `CoffeeScript` and trans-compiled to `JavaScript` that can run in any browsers. This makes BundleVis accessible as long as the users have a browser, which means BundleVis is cross-platform, and can run on both Android and iOS devices. Accessibility is important because non-expert users have started moving off from desktop computers and mainly use smartphones. Under the hood, BundleVis leverages D3.js graphics library [BOH11].

In this chapter, I introduce basic elements of BundleVis and illustrate how it visualizes frequent patterns. When compared with existing frequent pattern visualizations, strengths of BundleVis are summarized. Finally, my case study shows the functionality of BundleVis.

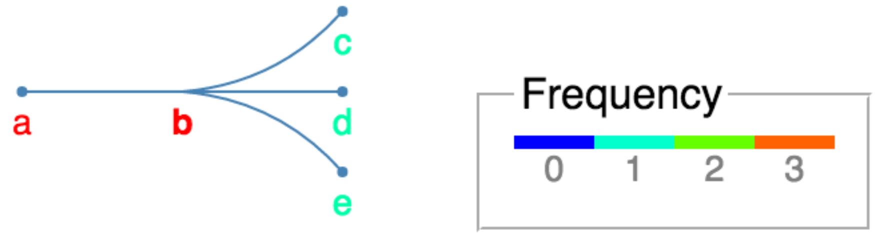


Figure 4.1: Basic representation of BundleVis and the associated color scale

4.1 Basic Representation

There are some basic elements needed to be presented in a frequent pattern visualization tool. In Figure 4.1, I show the visualization of a database of frequent items containing b: $\{a,\ b\}$, $\{b,\ c\}$, $\{a,\ b,\ d\}$, $\{a,\ b,\ e\}$. Here, I label domain items (e.g., a, b, c, d and e in Figure 4.1). The color of the label indicates the frequency of the item. The color scale comes with the visualization is also shown in Figure 4.1. The scale is from 0 to the maximum frequency in the domain. Note that the item of interest is placed in the center. Lines connecting the item of interest with some other items represent a frequent pattern. The frequency of a pattern is indicated by the lowest frequency in the line due to the Apriori property. For example, frequency of $\{b\}$ is 3, which is the same as the frequency of $\{a, b\}$. Solid dots means the itemsets can be expanded (e.g., from a 2-itemset $\{b, c\}$ in Figure 4.2 into a 3-itemset $\{b, c, e\}$ in Figure 4.3).

It is worth noting that the a-b-c line does not necessarily represents the frequent itemset of $\{a,\ b,\ c\}$. Recall that the visualization shows all frequent items containing b, which means that, in the line a-b-c, there must be frequent itemsets $\{a,\ b\}$, $\{b,\ c\}$ but it is not sure about $\{a,\ b,\ c\}$. I will explain further in subsequent sections.

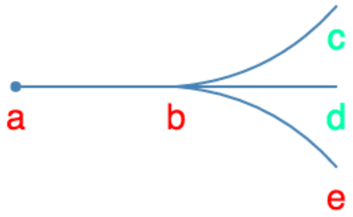


Figure 4.2: An example of BundleVis

4.2 A Simple Example

Now, I give an example in Figure 4.2 to show how does BundleVis handle frequent itemsets containing more than two items and how can users interpret and interact with them through BundleVis.

Before showing visualizations, users specify the frequent item of interest, called *split*. In Figure 4.2, the split item is b. With the split item, BundleVis first sorts frequent items in each frequent itemset, and only shows those frequent itemsets containing the split item and the two immediate neighbor nodes of the split item. In our example, item a is preceding the split item b and items c, d and e are proceeding the split item b. Because the initial view visualizes all frequent patterns containing the split item, the split item always has the highest frequency. After the initial view, users can select further items of interest from those immediate neighbor nodes. Non-immediate neighbor nodes are hidden.

To prompt users to expand or further select items of interest, non-solid dots are used. The concept is borrowed from mathematical notations: a solid dot indicates the end of the line. Users can click on any of the non-solid dots if the underlying item is of

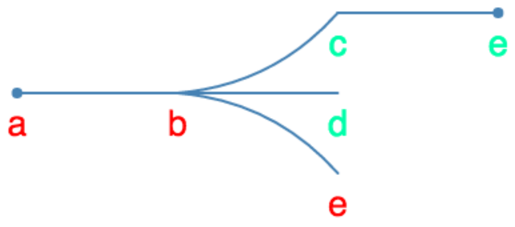


Figure 4.3: An example of BundleVis with c expanded

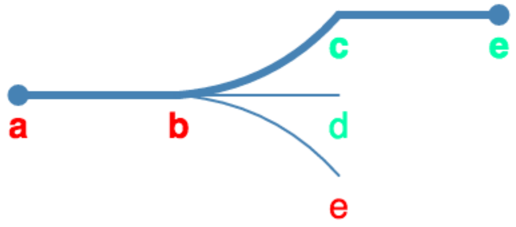


Figure 4.4: An example of BundleVis with c expanded and node e hovered

interest. Note that even all domain items are sorted before the visualization is shown, users can put domain items in any order by clicking blue nodes. This is superior to existing visualizations, as discussed in Chapter 3, where users have to understand what shown on the screen and filter information of interest, whereas BundleVis gives minimum information and users can freely explore. When a user wants to see what other items frequently associated with b and c, he can click on c. The expanded result is shown in Figure 4.3.

As shown, a blue dot representing e shows up. The blue dot indicates that this node cannot be clicked and expanded. This means there are no other items in frequent itemsets containing b and c where the other items are sorted after b. For the blue dot representing a, the interpretation is that there is no sorted items before a in the frequent itemsets containing a and b.

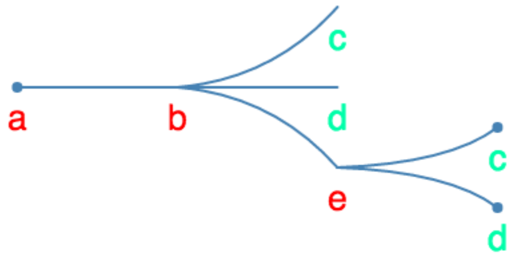


Figure 4.5: An example of BundleVis with c folded and e expanded

Again, from the line a-b-c-e, we know that there must be frequent patterns {a, b} and {b, c, e}, but we are not sure if {a, b, c, e} is a frequent pattern. Despite this, it is often a good idea to know the big picture to first get a rough idea and left details for further exploration until users want to. The big picture after is a and d each is frequently occurred with b and c. Users may be curious if {a, b, c, e} is a frequent pattern. To find out, users can hover on e, as this has already happened in real world when the user click on node c. The visualization when hovered on c is shown in Figure 4.4.

In Figure 4.4, the thickened line shows all frequent items containing b, c and e. Because line a–b is thickened, it answers the question that {a, b, c, e} must be a frequent pattern. As users know that {a, b, c, e} is a frequent pattern, they can also derive that any subset of the frequent pattern is also frequent (e.g., {a, c} and {b, e}).

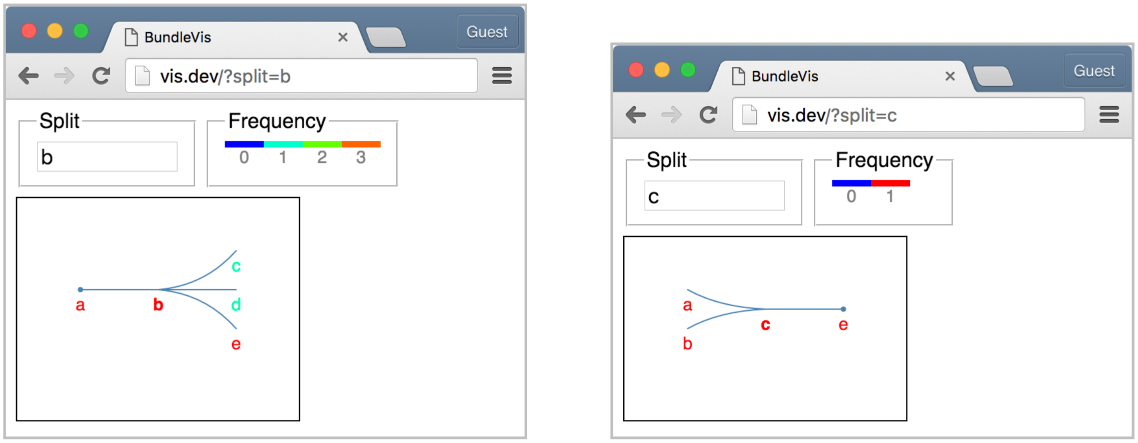
Now if users would like to know frequent pattern containing {a, b, e}, they can click on c to hide items that are not interesting anymore, and click on e. The result is show in Figure 4.5. Hence, users now know there are two frequent patterns containing b and e: {a, b, e, c} and {a, b, e, d}. Note that even the underlying data presentation is sorted, it is not visually sorted, i.e., the highlighted c is after e. This makes sure that the order of frequent items follows users' flow of thinking.

4.3 Distinct Features

4.3.1 Show the Vital; Explore at Will

When compared with FIsVis and other line-based frequent pattern visualizations (e.g., WiFIsViz and FpViz), BundleVis shows users the most important information in the first place by allowing users to choose interested domain items in the beginning. In FIsVis, nodes are depicted relative to the initial node that is the first node of the underlying sorted representation of a frequent itemset. For example, in Figure 3.1 and Figure 3.2, a is the initial node. This leads its disadvantage that the first domain item after sorting is what a computer preferred but hardly a person. It is not uncommon that the first sorted item is of trivial interest. In BundleVis, users can choose the item of interest in the very beginning of their exploration, i.e., the split that was mentioned in Section 4.2. In contrast, given Figure 3.1 and Figure 3.2, if users are interested in c, they cannot comprehend useful information from those line-based visualization.

In BundleVis, users can specify the split in two ways. One way is tech-savvy that one can change the split by changing query string `split` in the URL, as shown in the address bars of Figure 4.6. The current split in the left subfigure is c. Changing `split=c` to `split=b` will lead to the right subfigure of Figure 4.6. A more user-friendly way is to type the split item in the split input field that is shown at the top left corner of Figure 4.6. Users can then press enter key to finish. If b is typed,



vis.dev/?split=b

Split: b

vis.dev/?split=c

Split: c

Figure 4.6: BundleVis: Left has *b* as the split item. Right has *c* as the split item

pressing enter key will also lead to Figure 4.6.

4.3.2 Empower Users: Multi-tasking

As shown from Figure 4.1 to Figure 4.6, the visualization is shown in one browser tab. When users want to examine different series of frequent patterns, they can open existing tabs while leaving old tabs unaffected. As such, users can switch between to get more information. Users can also use those features that browsers have to offer. For example, users can put each tab in a separate window or even separate monitors to examine visualizations. Users can also leverage browsers' built-in features, such as "find in page" to find items and zooming to further focus on specific items.

Except browser tab level multi-tasking, one can examine frequent patterns in a single tab or visualization. Recall that a blue node can be clicked to get expanded and folded. This means users can leave multiple nodes expanded in order to compare or make the visualization more informative. An example is shown in Figure 4.7, where

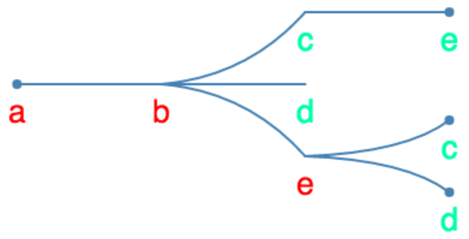


Figure 4.7: BundleVis where multiple nodes (i.e. c and e) are expanded

c and e are expanded at the same time. In this case, b was chosen as the split item. c is then clicked in order to examine frequent patterns containing b and c. Users then found e and wanted to know what would be the frequent items containing b and e. Users then clicked e indicated by the blue dot.

Compared to FpMapViz and RadialViz, where users have the option to zoom in and out, users cannot do multitasking at the same time. The disadvantage of zooming is that, after zooming in, users are locked in those frequent patterns previously selected, and users have to zoom out to get back in order to explore frequent patterns without any of those items previously selected. In the worst case, when users are at the point exploring $\{b, e, c\}$, and they would like to see frequent patterns with $\{e, c\}$ without b, users then have to go back to $\{b, e\}$, $\{b\}$ and $\{\}$, and then go to $\{e\}$ and $\{e, c\}$, which consists of five tedious tasks. In contrast, BundleVis empowers users and is more enjoyable to use: they can just click e and c, which reduce the tasks to a half effort.

4.3.3 Mobile Usage

Because BundleVis is implemented in JavaScript, a programming language that all browsers support, it can be run on any devices that have a browser. This makes BundleVis cross platform and can be viewed not only on desktop computers but devices with smaller screens such as smartphones or tablets. This is important as users have started using cellphones more than desktops. Compared with existing visualizations surveyed in chapter 3, their implementations only allow the visualization tools run desktop and are thus limited to one platform.

4.4 Case Study: A Course Dataset

In this section, I go through a case study on a course data set. It illustrates how a real world data exploration can benefit from visualizing with BundleVis.

The dataset is recorded in a computer science course CS 438. There are 50 students enrolled in the course and thus the dataset captures 50 student records. Each record consists of other courses the student is taking together with CS 438 in the current semester. One example record would be {402, 438, 458, 472}. This record represents a student who takes CS 438 and is also taking CS 402, CS 458 and CS 472. The raw dataset and the ID-description mapping for each course are appended in Appendix A. In the dataset, each line represents a record and items in a record are separated by a space.

Having described the dataset, we now explore some questions that BundleVis can

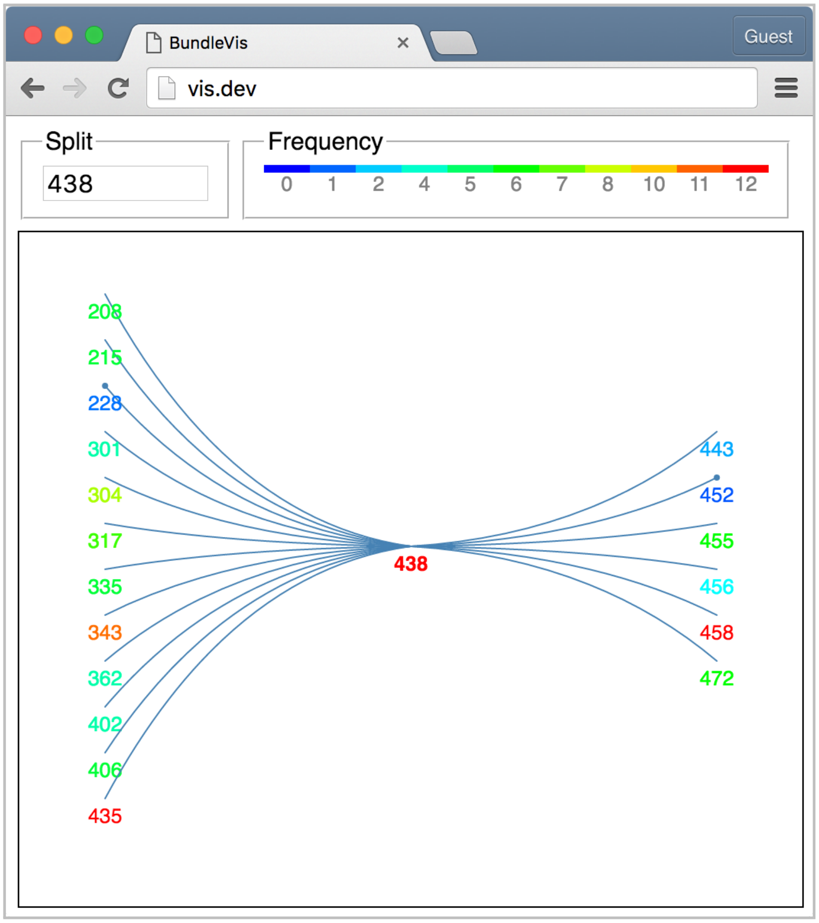


Figure 4.8: Intial visualization for the course dataset

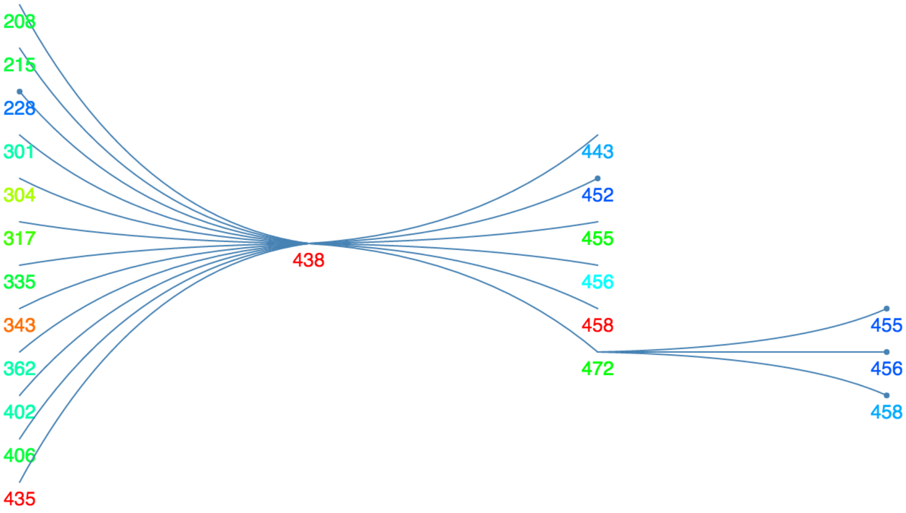


Figure 4.9: The visualization for the course dataset after 472 is expanded

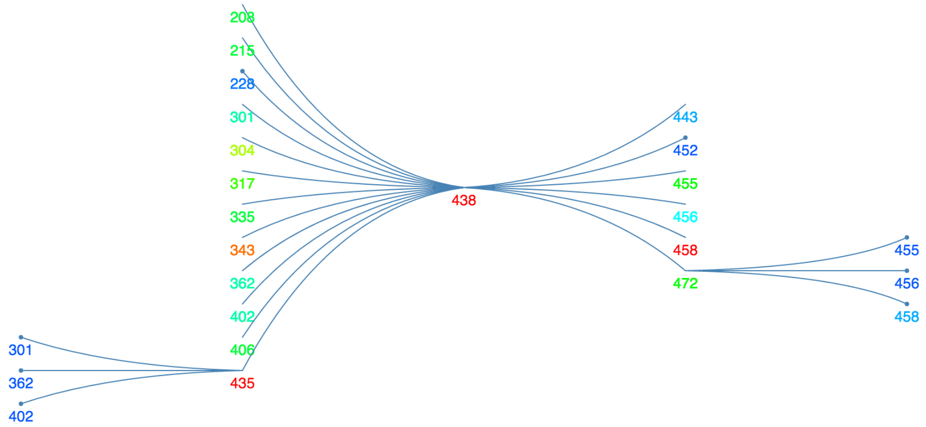


Figure 4.10: The visualization for the course dataset where 472 and 435 are expanded

assist answer. Figure 4.8 shows the initial visualization.

Imagine a Data Mining Professor X taught Database Implementation CS 438 and Data Mining course CS 472. He may want to know what the other courses his students are also taking are. To answer this question, he can easily click the 472 node to see some of the courses co-taken by students. As shown in Figure 4.9, the courses are CS 455, CS 456 and CS 458.

Some questions can also be answered immediately after Professor X sees the initial visualizations. What is the highest enrollment of other courses for my students? Courses in warmer colors (i.e., orange and red) are the answer, i.e., CS 343, CS 435 and CS 458. However, what the lowest enrollment are? Courses in blue are the answer, i.e., CS 228, CS 452. He then can derive that while students are taking fourth-year courses, they may also still taking second-year courses: CS 208 and CS 215 are not uncommon as they appear greenish.

4.5 Summary

Visualization is important for non-expert users to comprehend frequent patterns. In this chapter, I introduced BundleVis, a novel tool that I developed to visualize frequent patterns.

In BundleVis, frequent patterns are represented by bundled lines. Initially, users choose an item of interest, called split item, and lines representing immediate nodes of the item are bundled and shown to users. The text of each domain items is colored based on their frequencies, from blue to green to red; a complete color scale is given. At line end, a solid node indicates that the line shows all frequent items after the split item; non-solid nodes implies that users can still click on it and that the line does not show all frequent items. To help readers to easily comprehend the basic presentation, a simple example is given. I also went through a case study on a course dataset to illustrate a potential real-world use case of BundleVis.

Compared to existing frequent pattern visualizations, BundleVis visualizes the most interesting and important part to users and allows user explore as will and do multi-tasking. BundleVis allows users to specify the item that they are interested in (the most interesting part to users). This is very different to the line-based frequent pattern visualizations (e.g., WiFIsViz and FpViz), which put the first sorted domain item in the first place. Once users are in the visualization, only two levels of data (the most important part) is revealed: the two immediate neighbors of the item of interest. Users then can further explore the data at will by clicking on non-solid lines to see more items if there are any or hide them. The expanded node remains expanded when other items are expanded, which allow users do multi-tasking during interest shifts by users. Users can also leverage browsers' tabs to perform multi-tasking as well. Because BundleVis is implemented in a browser, users can also take advantage of other browser features such as “find in page” to find items and zooming to further focus on specific items.

Chapter 5

PatternShow: Scalable FPM System

In this chapter, I develop a user-friendly and scalable FPM application: PatternShow. PatternShow is a web application on the surface, but the backend is distributed and supported by cloud computing to make it scalable. Those two features are the concentrations while developing PatternShow.

In this chapter, I discuss the architecture of PatternShow that makes it scalable and talk about the web application that makes it user-friendly.

5.1 The Architecture

In order to make the PatternShow service scalable, service-oriented computing paradigm is adopted. PatternShow is decomposed to several services and each service can work independently. Figure 5.1 shows the complete workflow of PatternShow. It

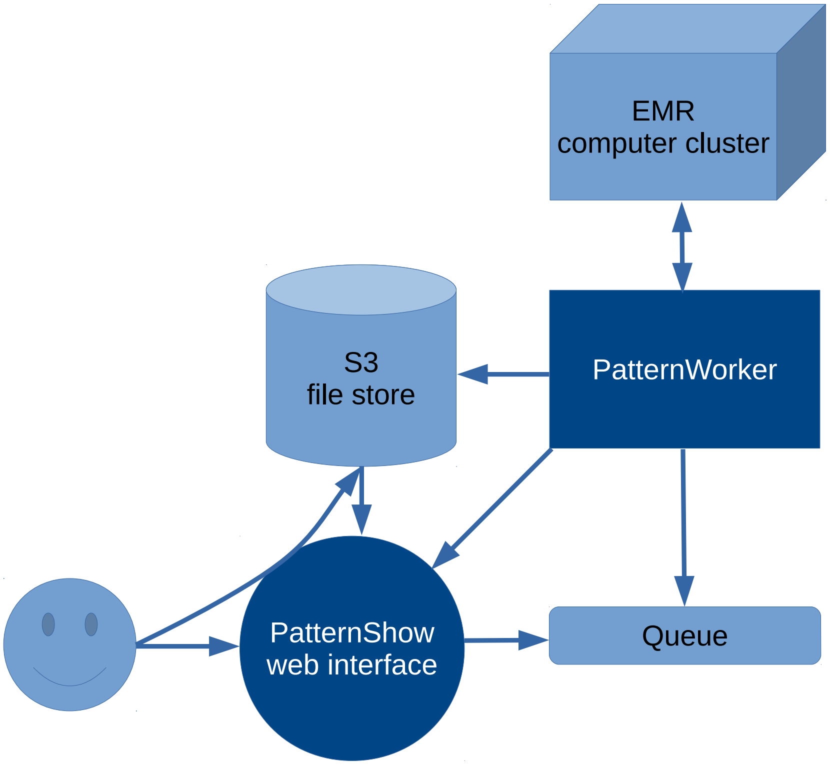


Figure 5.1: The architecture of PatternShow

can be read from left to right and bottom to top.

Here is an overview. Users register and sign in to PatternShow website and can upload data sets to the Simple Storage Service[1] (S3) file store directly through PatternShow. In the interim, S3 notifies PatternShow the upload progress and when it is finished. After a data set is finished uploading on S3, PatternShow does not process the data immediately by itself but create a job in a queue. PatternWorker, on the other hand, actively polls jobs from the queue and leverage Elastic MapReduce[2] (EMR) computer cluster to process data sets. Afterwards, PatternWorker puts the

[1] https://aws.amazon.com/s3/
[2] https://aws.amazon.com/elasticmapreduce/

file containing frequent patterns back on S3 and informs PatternShow. PatternShow finally visualizes the frequent patterns and show to users. Users also have the option to download the frequent pattern file.

S3 is offered by Amazon Web Service (AWS)[3]. S3 provides fast, reliable and safe transmission, unlimited storage and scalability. By leveraging S3, a user can upload a data set file as fast as the user's Internet connection. In big data era, it is not uncommon to see data sets ranges from gigabytes to terabytes. S3 can satisfy this requirement, allowing users to upload data sets as large as 5 terabytes. In PatternShow, data transmission is offloaded to S3. Data are directly transferred to S3 through users' browsers without interacting with PatternShow's web application server. The upload webpage is here only to generate and collect parameters, such as data file path and authentication information, which are needed to upload to S3. By doing this, PatternShow web server is not blocked by the file uploading process and, thus, not forcing other users waiting. Thanks to S3, PatternShow is then empowered to handle as many as concurrent uploads requested.

Data are transmitted to S3 through SSL-encrypted API (Application Programming Interface) endpoints using HTTPS, which ensures the *security* and *integrity* of the transfer. The S3 API endpoint is temporally authorized from S3 per upload request through PatternShow website, preventing malicious users to abuse S3 resource without authorizing first. During the transfer process, data cannot be retrieved without the encrypted key stored on S3 servers. In the meantime, data cannot be modified or tampered without notice on the client side, because each fragment of data send through SSL has an authentication code for the receiver to authenticate when the

[3] https://aws.amazon.com/

next fragment of data are ready to transmit.

After users uploaded data to S3, a job is queued rather than processed during the request. In other words, the data processing is asynchronous. This increases the throughput of the web interface, making it scalable. PatternWorker, an application that is constantly checking the job queue through API of PatternShow, will consume the jobs in the queue if any. PatternWorker itself does not process the data and output frequent patterns. Instead, it passes the jobs with necessary parameters to a cluster computing system to do the heavy lifting. When the computer cluster is mining for frequent patterns, the user waiting on the web page get notified for the mining process, such as "being processed". The job processing system, part of PatternWorker is implemented in Java and responsible for taking jobs off the queue and changing the status of a job (e.g., processed).

PatternWorker is built on top of Apache Spark[4], a general cluster computing system specialized on processing big data. Machine Learning Library[5] (MLlib) in Apache Spark is used for mining frequent patterns from data. The built-in algorithm (in MLlib) is a distributed version of FP-growth [LWZ+08], which partitions users' data across the compute nodes of a cluster to process the data in parallel. The computer cluster used by PatternWorker is hosted on Amazon's EMR for easy cluster management. With EMR, PatternShow leverages its cluster management provide by the Yet Another Resource Negotiator (YARN) to allocate and manage compute resources on master and slave compute nodes. EMR also offers tight integration with S3, the file store. Because EMR servers and S3 servers are in the same data center,

[4] https://spark.apache.org/

[5] https://spark.apache.org/mllib/

it helps decrease communication cost and maximize transfer speed in between.

The data processing system, part of PatternWorker, has a simple interface that only needs two pieces of information to function: the URI of the data file on S3 and the minimum count of a frequent product or product set (i.e., frequent patterns). When frequent patterns are mined for, the system stores it back to S3. The user waiting on the web page is then given options to either visualize frequent patterns or download the file. The download link is temporary and expires within a specific period of time to ensure nobody else can download users' data.

Last but not least, to achieve scalability, all components are independent to each other, and hence can be substituted for higher performance if needed. The S3 file store can be replaced as long as another file store is compatible with S3 API, such as DreamObjects[6]. The PatternWorker, along with the EMR computer cluster can be replaced as long as another frequent pattern mining tool can check, remove, change the status of tasks from the job queue through PatternShow's API as well as respect a specific data format and put the mined frequent patterns onto S3. Indeed, Spark's frequent pattern mining algorithm can also be replaced by another algorithm as long as it respects the input data format and output data format.

5.2 The Web Interface

The user-friendly interface is a web application hosted on the cloud. The web application can be accessed everywhere and users can use it to mine frequent patterns wherever the data are located. For example, users can choose and upload data sets

[6] `https://www.dreamhost.com/cloud/storage/`

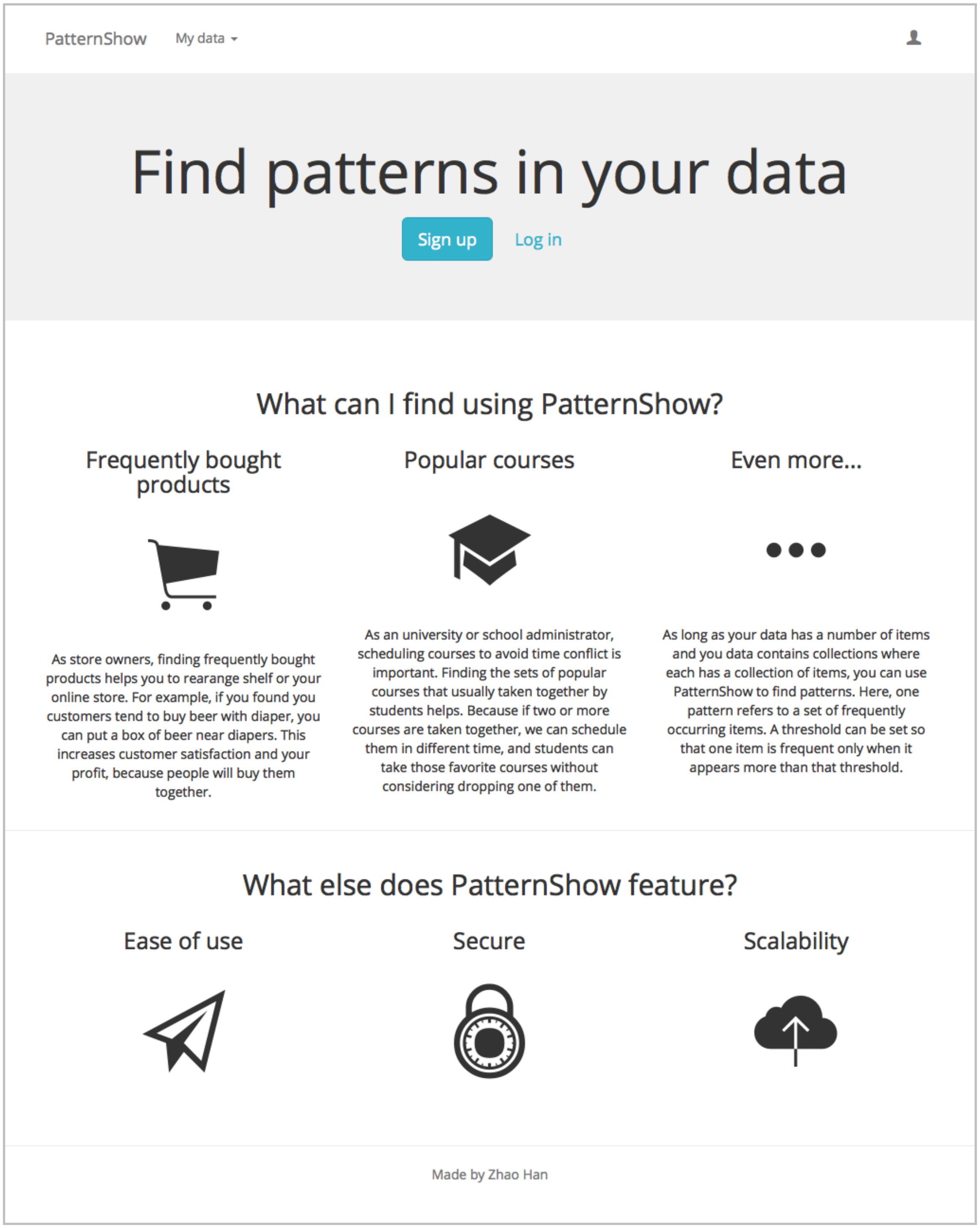
stored in their Dropbox accounts.

Because the web application is only responsible for user management and providing user interface, it does not require much compute power. Indeed, the web server used during development is the t2.micro instance[7] provided by Amazon Elastic Compute Cloud (EC2)[8]: it has 1 GB memory and runs on a 0.25 GHz CPU with the capability to burst to 2.5 GHz for 6 minutes per hour. As data transfer is offloaded to S3 without interacting with the web application, and frequent pattern mining is offloaded to EMR computer cluster, the web application, including the web server software, only consumes half of memory (450 MB) and a small fraction of CPU cycles (2%, 0.05 GHz) when idle. During the load test, it can handle 300 requests per second, which is sufficiently good as a starting point.

One of the most important components of PatternShow is user management. Next, I show how users will sign in and log in using screenshots. Figure 5.2 shows the homepage of PatternShow. In the wide gray area on the home page, users can click the "Sign up" button to register an account or click the "Log in" button to sign into PatternShow. Figure 5.3 shows the sign-up page where a user just needs to type email and choose a password to sign up. Figure 5.4 shows the login page where users type the same information in order to sign in. On the sign-in page, users can select "Remember me" so that they do not need to type the email and password for later visits. If users forgot their password somehow, they can easily reset the password by clicking "Forgot your password?" on the bottom on the sign in page. I implement

[7] https://aws.amazon.com/ec2/instance-types/#general-purpose

[8] https://aws.amazon.com/ec2/



Find patterns in your data

What can I find using PatternShow?

Frequently bought products

As store owners, finding frequently bought products helps you to rearange shelf or your online store. For example, if you found you customers tend to buy beer with diaper, you can put a box of beer near diapers. This increases customer satisfaction and your profit, because people will buy them together.

Popular courses

As an university or school administrator, scheduling courses to avoid time conflict is important. Finding the sets of popular courses that usually taken together by students helps. Because if two or more courses are taken together, we can schedule them in different time, and students can take those favorite courses without considering dropping one of them.

Even more...

As long as your data has a number of items and you data contains collections where each has a collection of items, you can use PatternShow to find patterns. Here, one pattern refers to a set of frequently occurring items. A threshold can be set so that one item is frequent only when it appears more than that threshold.

What else does PatternShow feature?

Ease of use

Secure

Scalability

Made by Zhao Han

Figure 5.2: PatternShow homepage

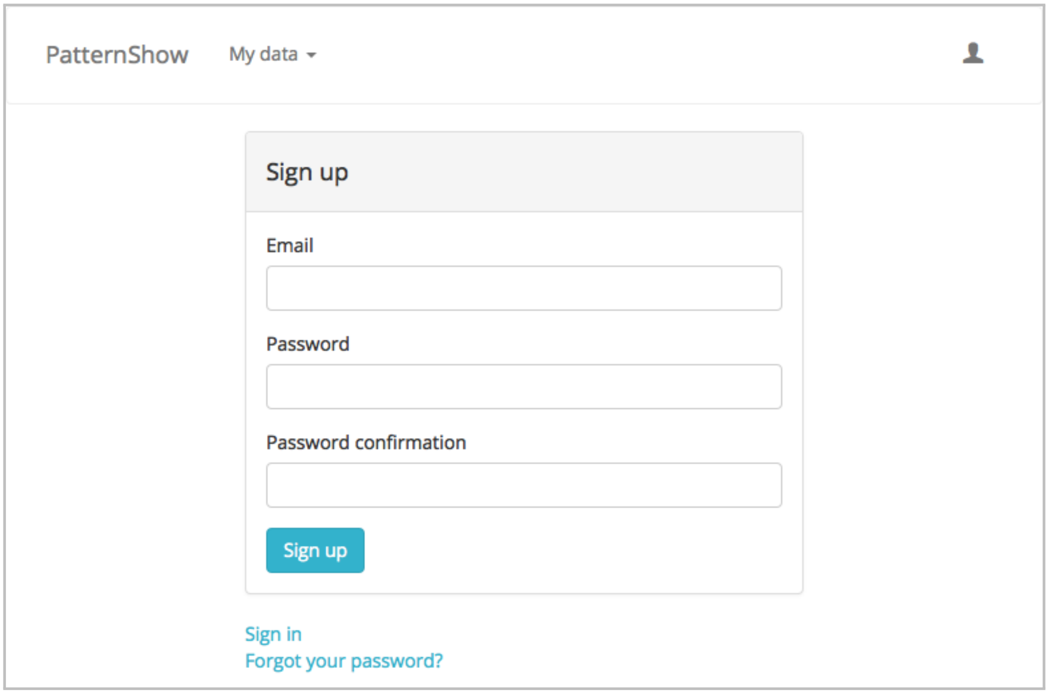


Figure 5.3: PatternShow sign up page

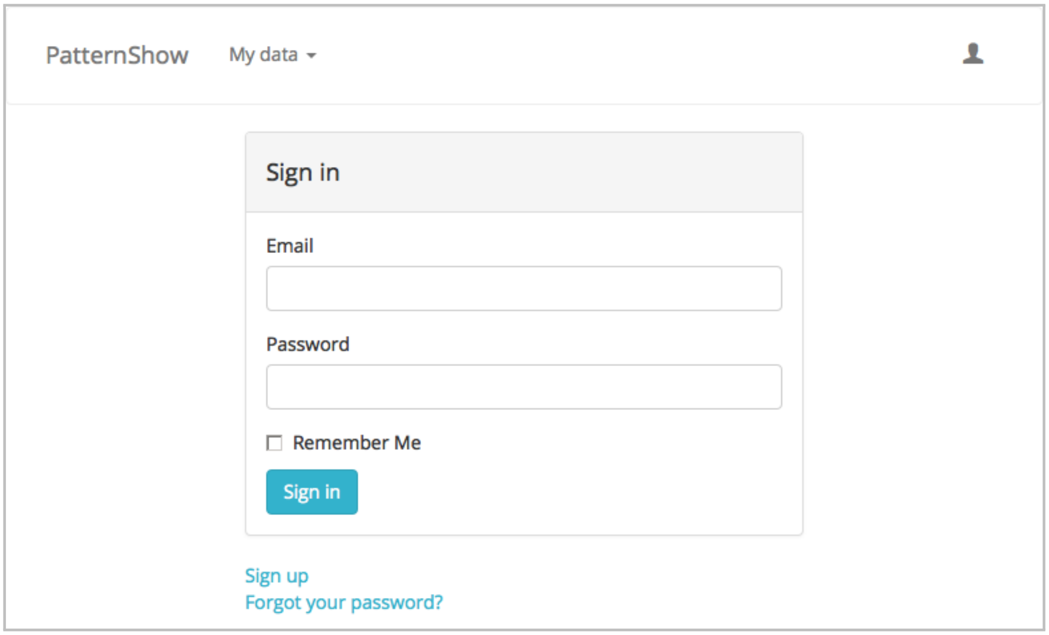


Figure 5.4: PatternShow log in page

those features because they not only are must-haves for websites but also meet users' expectations. Thus, they make PatternShow easy to use in the beginning.

After users successfully logged in, they will be redirected to a page to upload their data, as seen in Figure 5.5 on page 55. Alternatively, users can choose to see all of their previously uploaded data, by clicking the bottom link of "See all my data" as shown in Figure 5.5. Figure 5.6 shows the web page after clicking.

When users' data are uploaded, users take the first step to start their journey: set minimum support. As shown in Figure 5.7, users can type in a decimal number in the minimum support field. Below the input field, there is help note in gray to explain the minimum support concept to users.

After the minimum support is set and the data are processed, users will be asked to input their item of interest (as shown in Figure 5.8), which was detailed in the previous chapter.

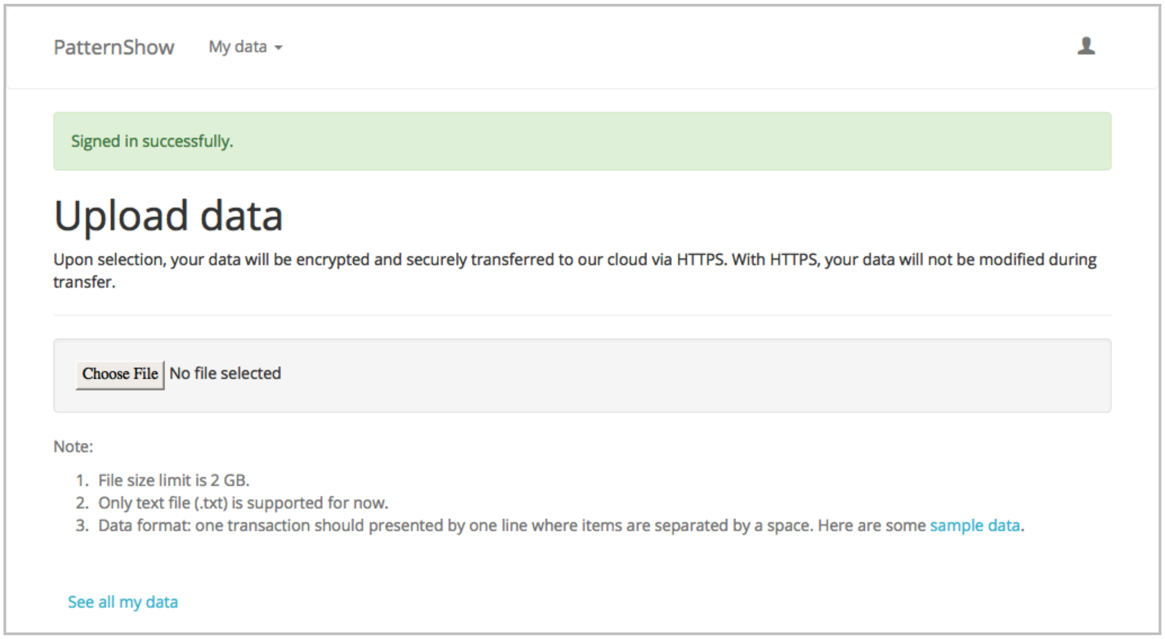
5.3 Summary

Before using any visualization tools to explore data, friendly user interface and a scalable backend are important to end users. In this chapter, I introduced PatternShow.

The architecture of PatternShow was discussed using a flow chart as show in Figure 5.1. Users log in to the web interface and upload their data directly to Amazon S3, a file store, without communicating with PatternShow server. When uploaded to S3, it notifies PatternShow and PatternShow creates a job in a queue. PatternWorker monitors the queue and uses Amazon EMR cluster to mine frequent patterns from the data stored on S3. When successfully mined, PatternWorker notified PatternShow, and PatternShow directed users to visualization.

As shown, PatternShow is service-oriented, and each part of PatternShow is independent of each other. Together with S3 and EMR computer cluster, PatternShow is made scalable.

The web interface is primarily a user management web application. As expected by users, it allows users to register, login, remember password, retrieve password, upload data, and access data. The visualization part included in the web interface was discussed in Chapter 4.



Signed in successfully.

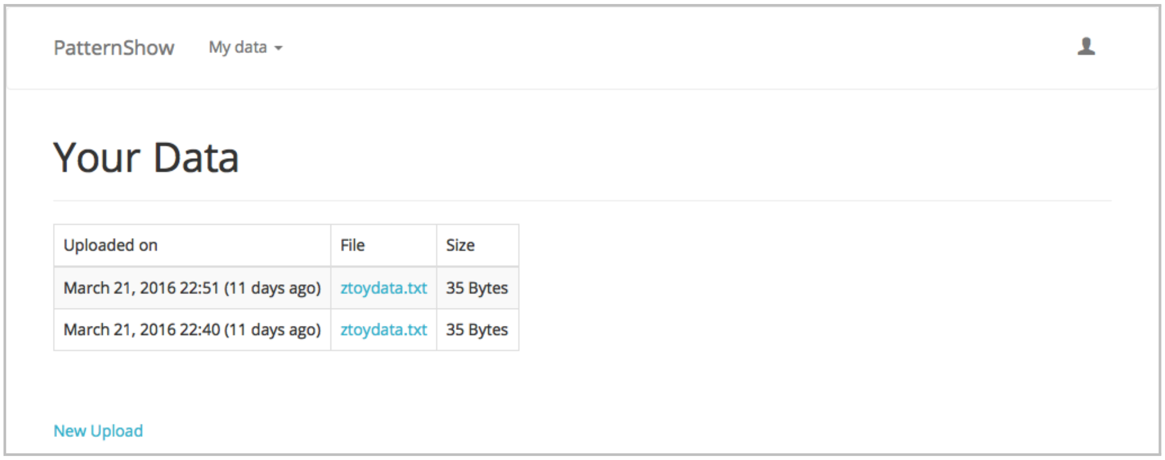
Upload data

Upon selection, your data will be encrypted and securely transferred to our cloud via HTTPS. With HTTPS, your data will not be modified during transfer.

Note:

1. File size limit is 2 GB.
2. Only text file (.txt) is supported for now.
3. Data format: one transaction should presented by one line where items are separated by a space. Here are some sample data.

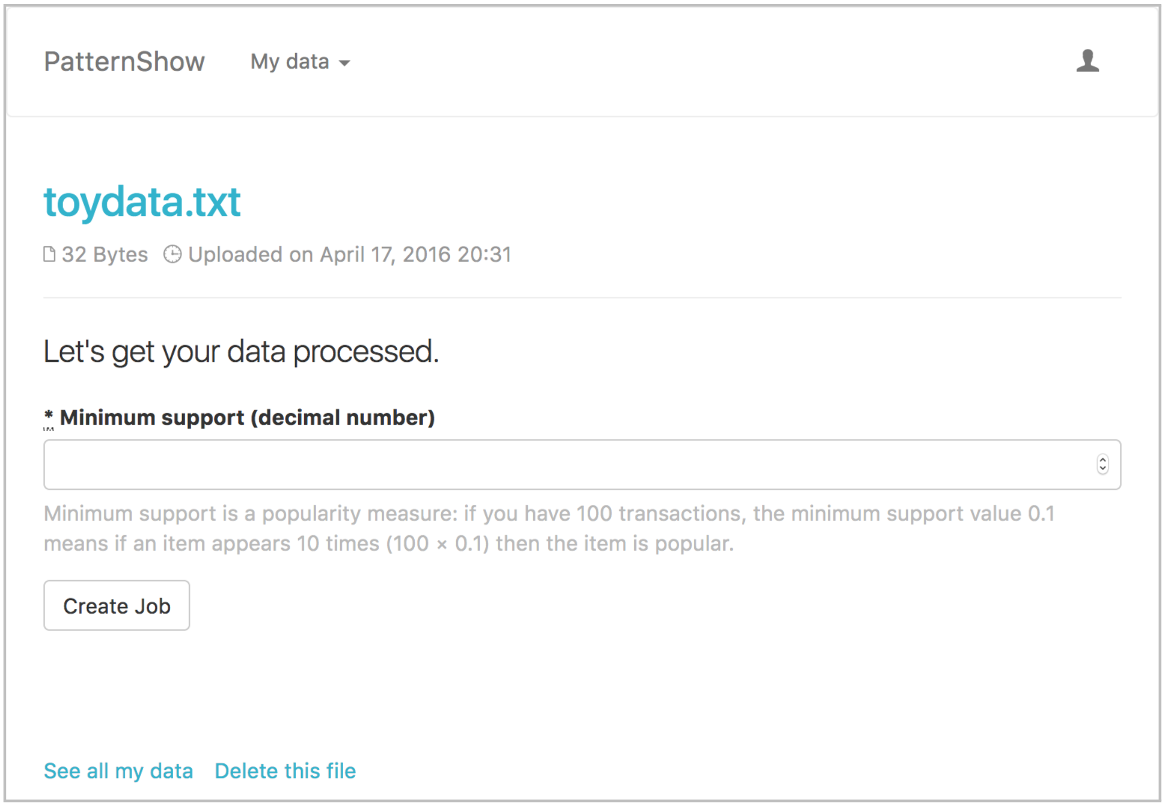
Figure 5.5: PatternShow upload data page



Your Data

Uploaded on	File	Size
March 21, 2016 22:51 (11 days ago)	ztoydata.txt	35 Bytes
March 21, 2016 22:40 (11 days ago)	ztoydata.txt	35 Bytes

Figure 5.6: PatternShow all uploads page

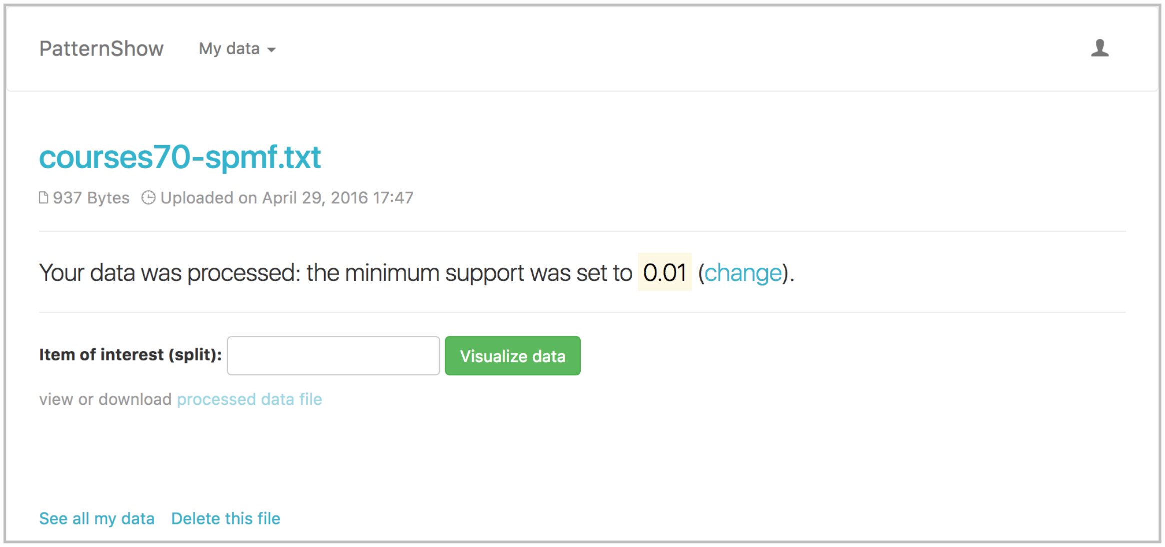


Let's get your data processed.

* Minimum support (decimal number)

Minimum support is a popularity measure: if you have 100 transactions, the minimum support value 0.1 means if an item appears 10 times (100 × 0.1) then the item is popular.

Figure 5.7: PatternShow minimum support page



Your data was processed: the minimum support was set to 0.01 (change).

Item of interest (split):

view or download processed data file

Figure 5.8: PatternShow item of interest page

Chapter 6

Evaluation

In order to verify the effectiveness of PatternShow's web interface and BundleVis (e.g., how easy or difficult to use it? how useful is it?), I designed a holistic user study and conducted the study with 11 participants. The user study mainly includes a set of tasks and a user satisfaction questionnaire. In this chapter, I will perform an extensive analysis based on the task completion time, the answers to the questionnaire, participants' comments and my observation on participants' behaviors. Potential improvements are discussed during the analysis. The result shows PatternShow' web interface and BundleVis is effective for non-expert data miners.

6.1 Participants and the Procedure

11 university students and recent graduates from various faculties participated in this user study: Four of them are expert users; the other seven participants are non-expert users. Even though PatternShow is designed for non-expert mines, experts'

data are also collected because, as discussed in the related work on FPM applications, PatternShow also benefits expert miners. In addition, it would also be interesting to see expert miners' reaction and opinion.

Before the user study starts, I present the participants with an informed consent form. We then divide the participants into two groups (expert data miners vs. non-expert data miners) by asking them to fill out a questionnaire, where participants answer if they consider themselves as an expert data miner, if they know data mining and its definition, if they know frequent pattern mining and its definition. The definition is asked on purpose to draw a clear distinction between experts and non-experts. The questionnaire is appended in Appendix B.

The remaining evaluation consists of a set of seven tasks and 1 user satisfaction questionnaire with seven questions. The whole evaluation was designed to take at most 46 minutes (a maximum of 5 minutes for each task and, for the questionnaire, 1 minute for the first six questions and 5 minutes for the last question). The tasks and the questionnaire are also appended in Appendix B.

The time to complete each task is recorded. For each task, there is a timeout for 5 minutes. The timeout was designed as an indicator of unacceptable performance of PatternShow. While completing each task, participants are asked to "think aloud": say what they are thinking and doing (e.g., I'm going to do... I'm looking for... I'm stuck...). Participants are prompt when they do not actively "think aloud". The "think aloud" technique gives insights on what the participants are thinking and helps evaluate PatternShow through later analysis. Through participants' talk and my observation, I take brief notes to record interesting events (e.g., "Oh! They are using the built-in search feature of the browser"). During the evaluation, a screencast (i.e., a digital recording of the computer screen capturing user input and computer output) is also taken for each participant without any sound records.

After completing all tasks, participants were asked to complete the user satisfaction questionnaire. The questionnaire mainly consists of closed questions. Participants are encouraged to leave comments in the end of the questionnaire for anything they did not speak while completing the tasks. The questionnaire is also appended in Appendix B.

Each participant completes all the tasks and answers all the questions. The University of Manitoba Fort Garry campus joint faculty research ethics board approved this study.

6.2 Dataset

A retail dataset from the Frequent Itemset Mining Dataset Repository Website[1] is used. The repository is famous in the frequent pattern mining academic community: researchers in the field did an extensive research using the datasets in the repository [MAG13, ZZB11].

The dataset includes 88,163 anonymized retail store transactions from an anonymous retail store in Belgium. A sample record is {8, 36, 38, 39, 41, 48, 79, 80, 81}; each number is an identifier for a merchandise item.

In the task description, I associate each product identifier with a random product name such as banana or apple. This helps users to easily understand the individual

[1] http://fimi.ua.ac.be/data/

record so that they can focus on the tasks.

The minimum support was set to $\frac{8}{88163} \approx 0.00009$ in this study. This means that, if there is a merchandise item appearing more than 8 times, the item is considered to be frequent. The value is small because large minimum support value tends to eliminate useful information that user may need. For example, if the minimum support value is set to a larger value 0.009, only those products being bought 794 times will be considered frequent, while users may want to see products that appear 600 times. Despite this discussion, the minimum support value itself is not the focus of this user study, as the data and the interface, including the visualization, are where users mostly interact with and, thus, what I want to evaluate.

6.3 Result Analysis

I analyze the evaluation results based on a combination of task completion time data, user satisfaction questionnaire answers, and the notes I took through participants' "think-aloud" process and my observation. Along with the analysis, potential improvements are also discussed.

Before getting started, it is worth putting the emphasis on completion time. Completion time is an important variable because it measures both the efficiency and effectiveness of PatternShow. The aforementioned 5-minute timeout indicates effectiveness: if a participant takes more than 5 minutes to complete one task, PatternShow is not effective at the task; otherwise PatternShow is effective at that task. The completion time is also mapped to efficiency: the less time participants spend on a task, PatternShow is more efficient on the task.

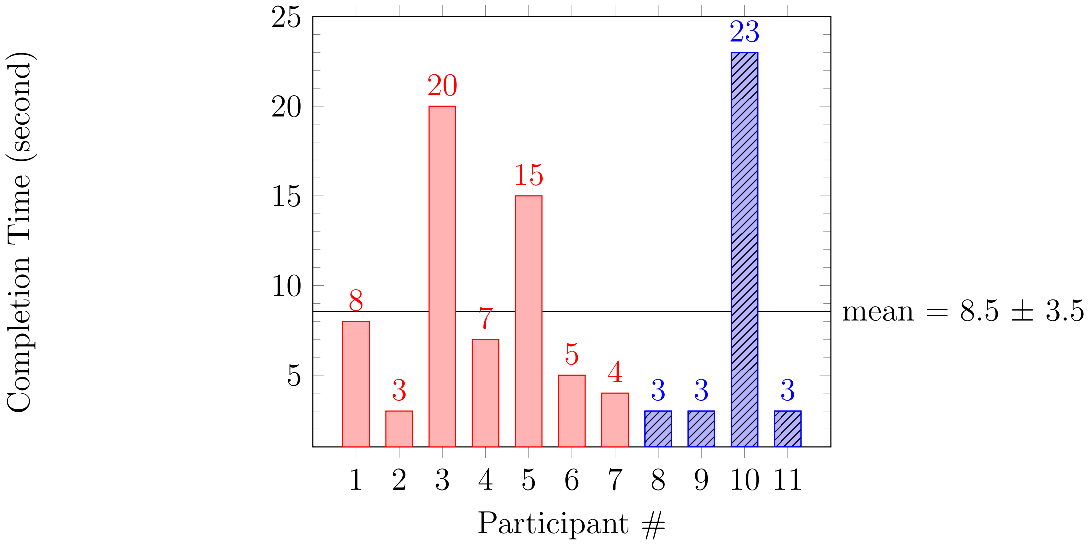


Figure 6.1: Completion Time for Task 1: Please upload the data file retail.txt located on desktop.

6.3.1 Result Analysis on PatternShow

The first task is “Please upload the data file retail.txt located on desktop”. As expected, most participants (8/11, 73%) complete this task under 10 seconds with mean value 8.5 seconds, shown in Figure 6.1 where Participants 8, 9, 10 and 11 are expert miners and indicated by blue patterned bars. Some participants spend longer on this task because they read the note on the upload page, a screenshot of which can be found in Figure 5.5 on page 55.

The second task is “As stated before, there are 88,163 transactions, please fill up the minimum support field (In this study, we assume if there are eight transactions containing an item, then the item is frequent)”. This was mainly designed to evaluate if users can understand the fundamental concept, minimum support, with the help

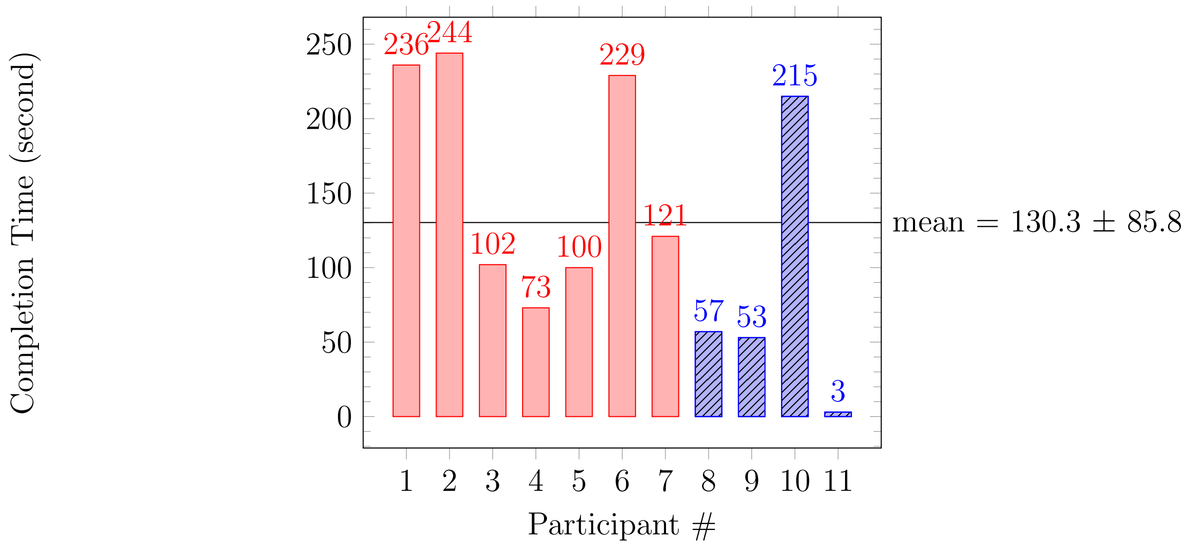


Figure 6.2: Completion Time for Task 2: As stated before: there are 88,163 transactions, please fill up the minimum support field (In this study, we assume if there are 8 transactions containing an item, then the item is frequent)

note that is shown in Figure 5.7 on page 56.

The completion time for this task is plotted in Figure 6.2. As shown, three (27%, Participants 8, 9 and 11) participants spend less than 1 minute to fill out the minimum support value; four (36%, Participants 3, 4, 5 and 7) participants spend more than 1 minute but not more than 2 minutes; four (36%, Participants 1, 2, 6 and 10) participants spend around 4 minutes to finish this task. Seven of the 11 participants (64%) completed the tasks in 2 minutes.

Before interpreting, please recall that Participants 8, 9, 10 and 11 are expert miners. Participants 8, 9 and 11 are among those who spend less than 1 minute to finish the task. Participant 10 spends more time reading the information presented

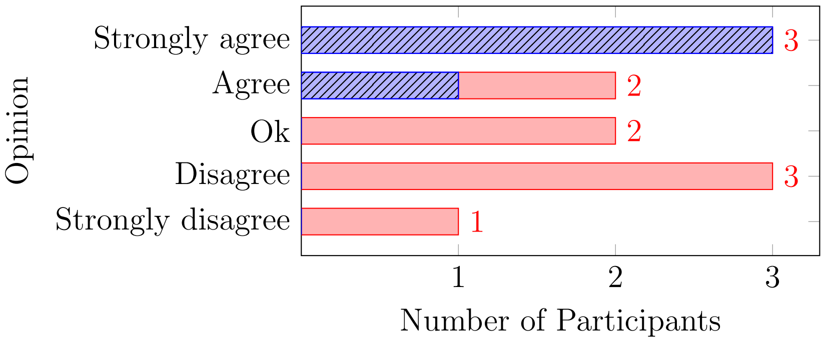


Figure 6.3: Answers to Question 1 in User Satisfaction Questionnaire:

The help note for "minimum support" is helpful to understand the concept.

on the web page.

In general, participants did not expect to do any calculation in this evaluation. Most non-expert miners complain the calculation even though they are solving a simple linear equation. This is not only reflected by the completion time but also by the corresponding question in the user satisfaction questionnaire. As plotted in Figure 6.3 where red bars indicate expert users and blue patterned bars indicate non-expert users, three participants (27%) checked "Disagree" when asked if the help note for "minimum support" is helpful to understand the concept and 1 participant checked "Strongly disagree".

In retrospect, the input field should simply be "how many times that an item appears should be considered frequent?" rather than asking users to solve a linear equation: "The total number of items × minimum support value = number of times a popular item appears". The simpler question does not need users to understand a new concept nor do any calculation. The concept "minimum support" should also be left as an advanced feature and put in help note section.

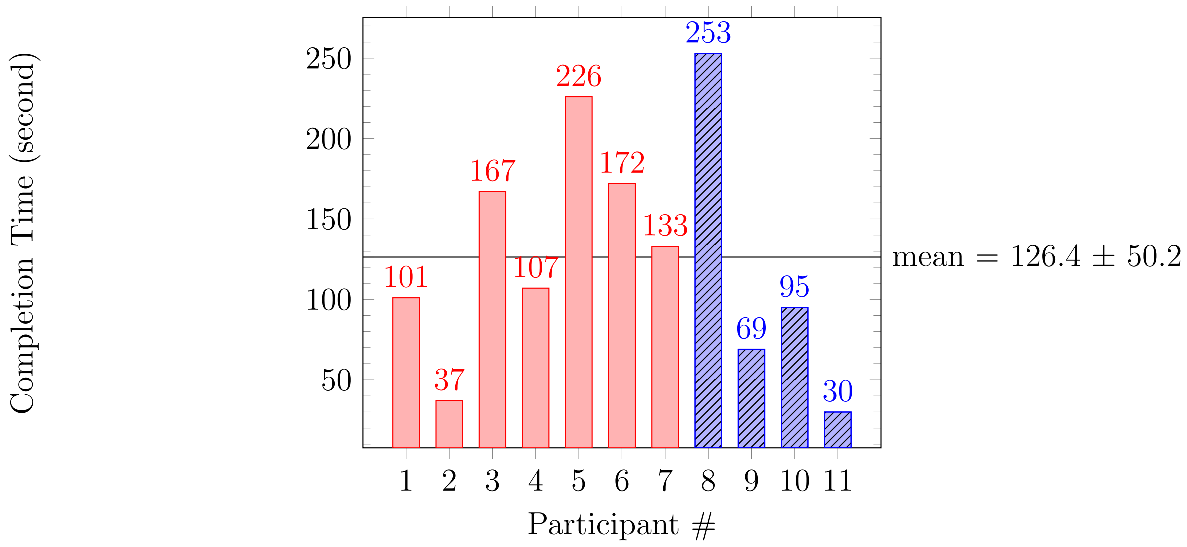


Figure 6.4: Completion Time for Task 3: What are the most popular merchandise items that customers usually bought with item 1198 (apple)?

Participants also tend to ignore the help note in the first place. Users only start reading the help note when they found they are stuck, which was indicated by users saying "I don't know what is this?", "Um... Should it be eight? No...".

Despite this, PatternShow is effective because all participants eventually know how to calculate the minimum support value within 4.5 minutes while seven participants complete the task within or around 2 minutes. In addition, five (64%) participants agreed that the help note is helpful and two participants said it is ok.

6.3.2 Result Analysis on BundleVis

The third task is "Which are the most popular merchandise items customers usually bought with item 1198 (apple)?" This task officially introduces users to BundleVis that visualizes frequent patterns. The completion time is plotted in Figure 6.4.

The mean time to complete this task from participants is around 2 minutes. However, there are four participants spending more than 3 minutes to complete the task. I have a few thoughts on the reasoning behind this. Recall that users are presented with the item of interest input field, shown in Figure 5.8 on page 56. From what I have observed, participants did not expect to reveal the answer by typing in the item of interest, an extra step. Two participants tried to find the answer by checking the "processed data file" link. Only when they found the long list of textual result is impossible to interpret, the participants click the back button in the browser. In retrospect, Task 3 should have been separated into two tasks. The task presented to users first should be that, in order to know which are the most popular merchandise items customers usually bought together with item 1198 (apple), what will be your item of interest? Task 3 then becomes the fourth task.

Another observation is that two participants tried to click "visualize data" button directly, ignoring the item of interest input field. Even though the completion time data shows the effectiveness of PatternShow, an improvement is to have a default item of interest value, such as the most frequent item in the transactions. Another two participants tried to type apple, the name of the item, instead of its identifier 1198 in the input field. This leads to the improvement that the backend may find the identifier based on the item name. All of these contribute to the total time to complete the task.

Despite this, when asked in the user satisfaction questionnaire if participants find it useful to specify the item of interest, most participants (10/11) answered agree (8/11) and ok (2/11) while only one disagrees. The data is plotted in Figure 6.5.

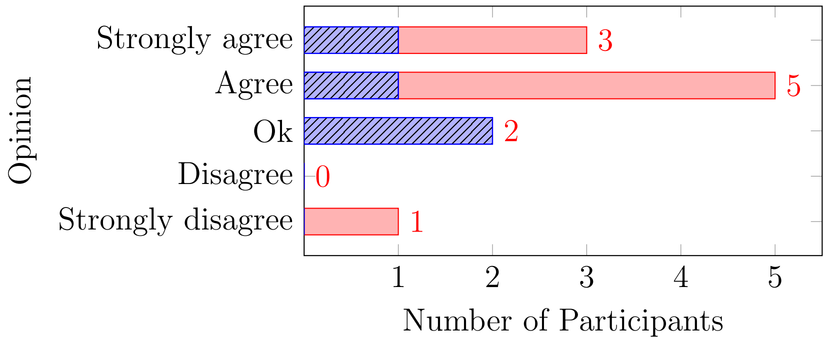


Figure 6.5: Answers to Question 2 in User Satisfaction Questionnaire: I find it useful to specify the item of interest.

After users get into the visualization page, several behaviors from participants attract my attention before the participants find the items that are frequently bought together with item 1198 (Apple). Once on the visualization page, more than half participants ignore the information located on top, such as the color scale for frequency and the legend telling users what can be expanded and what cannot. After participants finish this task, they complained that there should be more explanation. When I tell them that there is actually a simple example for this visualization in my thesis writing, they agree that it helps if they read a simple example. Some participants also explicitly asked for more explanation material on the webpage or even an introductory video. I stayed neutral while answering those questions by saying “Yeah, that is a good idea””. I purposely did not explain the visualization while counting the completion time. In retrospect, a short tutorial or an introductory video can be shown on the homepage of PatternShow.

In the third questionnaire question, illustrated in Figure 6.6, no participants disagree that it is not easy to find the most popular merchandise items while 7 partici-

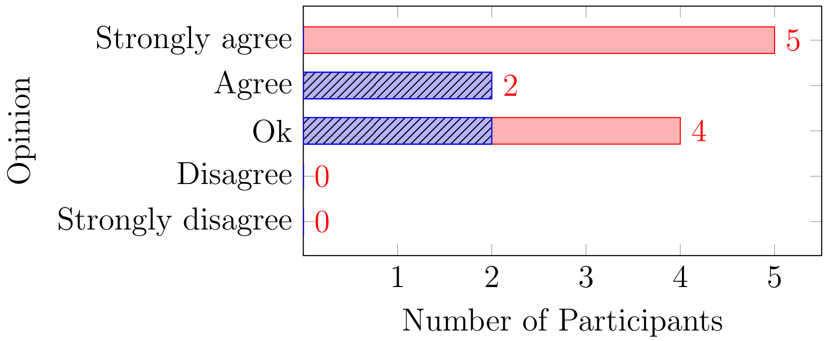


Figure 6.6: Answers to Question 3 in User Satisfaction Questionnaire:

It is easy to find the most popular merchandise items.

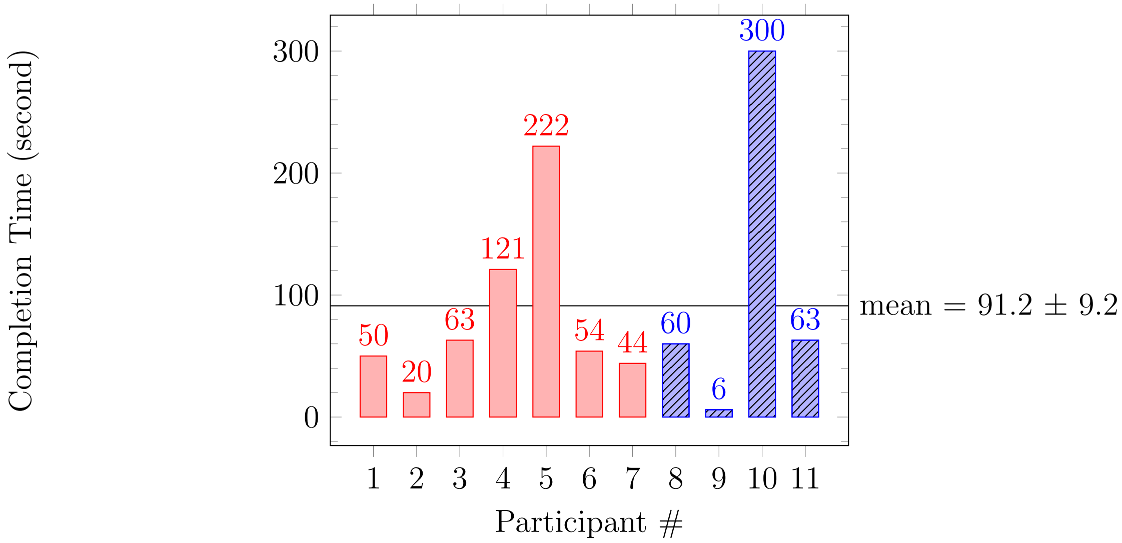


Figure 6.7: Completion Time for Task 4: How many customers bought both item 1198 (apple) and item 48 (banana)?

pants agree. Along with the second questionnaire question (Figure 6.5), the completion time and the answers show the effectiveness of PatternShow at this task.

The fourth task is "How many customers bought both item 1198 (apple) and item 48 (banana)?" The completion time is plotted in Figure 6.7.

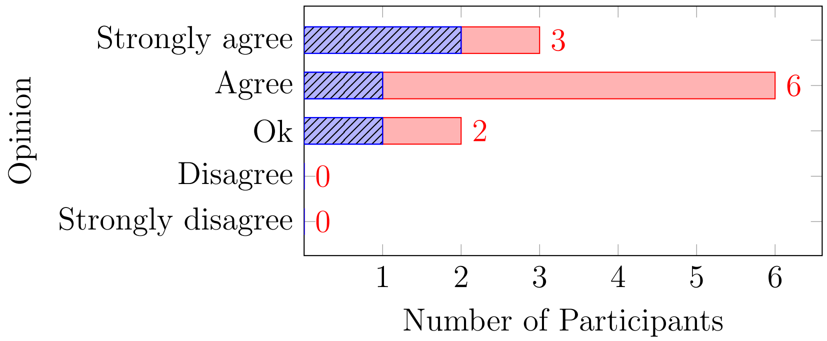


Figure 6.8: Answers to Question 4 in User Satisfaction Questionnaire:

It is easy to find how many customers bought both apples and bananas.

Participant 10 spends more than 5 minutes to finish the task, which leads to timeout. One of the reasons is that the participant had a hard time to interpret the tree representation: items connected by lines are not considered to be connected by the participant, because of which the participant cannot find how many customers bought both A and B. The participant also had a hard time to understand the frequency color scale: red color means small number. Because participant 10's behavior is very different from other participants, I considered the participant as an outlier.

Even though participant is an outlier, the participant does suggest that the frequency number mapped to a color is confused with the item identifier in the visualization. Two participants also expressed that they prefer product names instead of numerical identifiers. This is an opposite opinion to the academic world where numerical identifier is a common practice. It is reasonable to use numerical identifiers in research because numerical identifiers are friendly to machines, but it is important to realize they are not friendly to human. Showing item names is a better way to show items to users, non-experts.

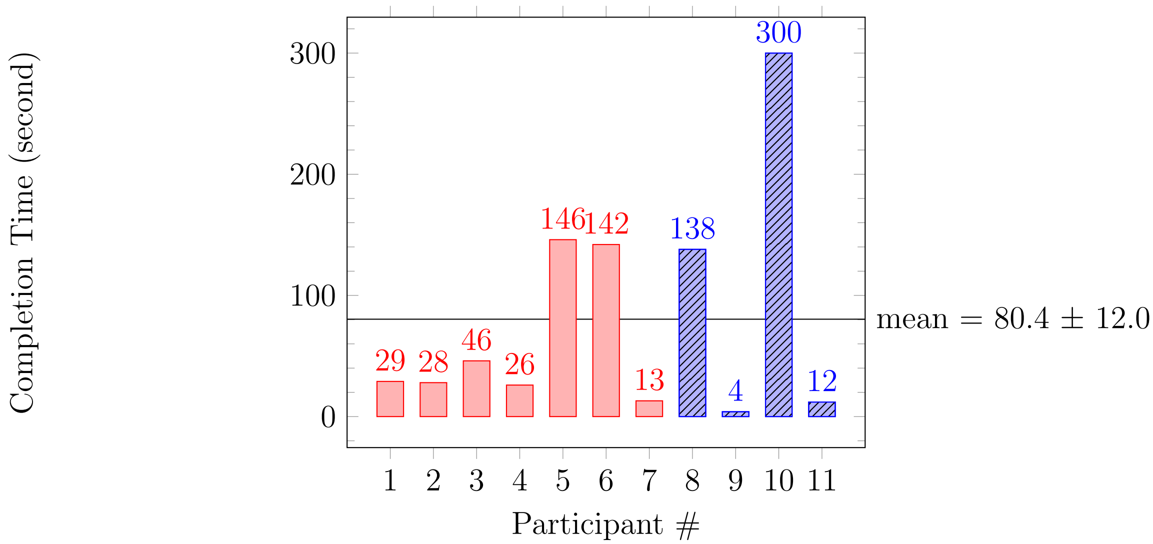


Figure 6.9: Completion Time for Task 5: How many customers bought item 1198 (apple), item 48 (banana) together with item 39 (cherry)?

In terms of effectiveness, most (9/11) participants complete this task within 2 minutes. Seven participants even complete this task within 30 seconds. Among those seven participants, four participants used browser's built-in search feature to quickly find item 48 (banana).

In the user satisfaction questionnaire, shown in Figure 6.8, no participant disagree that it is easy to find how many customers bought both apples and bananas. Along with the completion time data, PatternShow is effective and efficient for this task.

The fifth task is "How many customers bought item 1198 (apple), item 48 (banana) together with item 39 (cherry)?" This task is important because it evaluates the interactive feature of BundleVis: click to further select item of interest, which was discussed in Figure 4.3 on page 34. The completion time for this task is plotted in Figure 6.9.

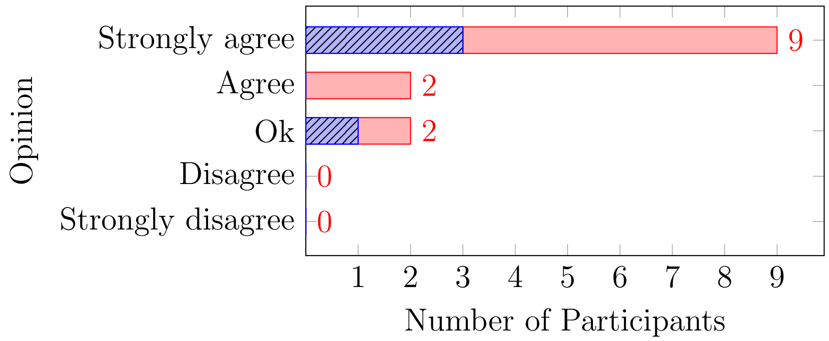


Figure 6.10: Answers to Question 5 in User Satisfaction Questionnaire: It is easy to find how many customers bought apples (item 10258), bananas (item 78963) together with cherries (item 15987).

Participant 10 again spends more than 5 minutes to complete the task, which leads to timeout for the second time. The participant had a hard time to recognize that an item node can be expanded. The participant kept ignoring the legend information on the top of the webpage. Even though he is the only one who did not figure out how to expand an item, an improvement on PatternShow can be an introductory video on PatternShow homepage or a section called" how to read the visualization" on the visualization page.

Three participants (Participants 5, 6 and 8) were confused if line end circle can be clicked on to expand or it is the line that can be clicked. Using line end solid circle to indicate line end in mathematics does not apply to web interface. When users see a circle, they are inclined to click on it instead of knowing that there are no more items in a frequent itemset. One improvement is to make solid circle indicate that the line can still expand while line itself indicates the line cannot be further expanded.

Some participants tried to use multiple items in the item of interest input field.

This is a remarkably realistic question when completing this task, because this task asks a question about two items. If participants can use multiple items in the item of interest input field, then the task will be completed quickly. One improvement is to allow users to specify multiple items of interest at once.

Around half participants asked how items are sorted during the task. When I tell them that items are sorted by letters from left to right, they expressed "no wonder". To improve this, the web interface might need to explicitly tell users this piece of information. However, if item names instead of numerical identifier are shown, then the problem does not exist in the first place.

The mean time to complete Task 5 is 1 minute and 20 seconds, while seven (63%) participants finish the task within 1 minute. From this, we can conclude that PatternShow is effective and efficient at this task. In the user satisfaction questionnaire, shown in Figure 6.10, nine (82%) participants agrees that it is easy to find how many customers bought apples (item 10258), bananas (item 78963) together with cherries (item 15987). This result enforces the fact that PatternShow is efficient at this task.

The sixth task is "Among the customers who bought item 1198 (apple), how many other customers bought both item 48 (banana) and item 39 (cherry)?" This is the same question as Task 5 except that it is asked in a different way. This task was designed to evaluate how newcomers (i.e., non-experts) interpret a tricky question that may appear in an exam for a data mining class.

As shown in Figure 6.11, because Participant 10 was timed out during the previous task and did not find that this task is the same as last one, that participant had another timeout.

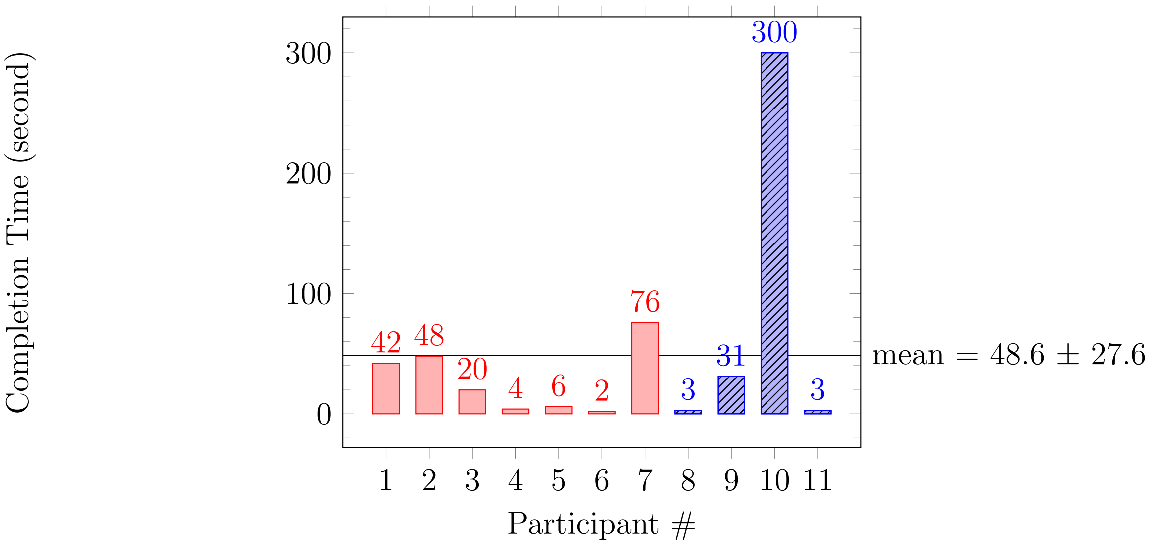


Figure 6.11: Completion Time for Task 6: Among the customers who bought item 1198 (apple), how many other customers bought both item 48 (banana) and item 39 (cherry)?

Five participants, i.e., 45%, (Participants 4, 5, 6, 8 and 11), quickly found that this task is the same as the previous task and, thus, only spend less than 10 seconds to complete this task. The remaining five participants did not tell the difference and thus spend no more than 1 minute and 10 seconds to finish this task.

From my observation and participants' use of the think-aloud technique, participant showed confused emotion but continued finishing the task by telling me the answer.

The last task is "What other information did you observe from the visualization?". It was asking participants' other thoughts while they do not forget yet. This task is accompanied with the last question in the user satisfaction questionnaire: "Do you have any other comments about the software that were presented to you (e.g., level of usefulness)?".

Some participants expressed that the visualization in a browser is very convenient because they can use the built-in find and zoom features. However, more than half users did not use browser's built-in features. One improvement is to inform users by putting a message above the visualization on the visualization web page.

Two participants expressed that the visualization might be more beneficial to business instead of consumers because consumers may spend more money for those items placed together but not what they want to buy.

One participant said that, as a consumer, the participant may not want to see frequently bought products together because the participant may want to discover products not in the shopping list.

The answers are remarkably interesting to researchers to think in the view of non-expert miners.

6.4 Summary

Through the analysis of the evaluation result, while there is space for improvement, PatternShow is effective and efficient. It is effective because, except for Participant 10, all tasks are completed within the timeout period, 5 minutes. It is efficient because participants only spend a maximum of average 2 minutes and 10 seconds to finish all tasks.

PatternShow can be improved on several aspects. The improvement is also the first to provide research insights on non-expert miners' behavior when using a frequent mining system.

Instead of asking users to calculate minimum support, we should simply ask users how many times that an item appears should be considered frequent. While users like the item of interest input, it can have a default value (e.g., the most frequent item) and allow users to type in multiple items. Item names rather than its numerical identifiers should be used to represent items. Before presenting users with the visualization, a simple tutorial or an introductory video can be given. To inform users that an item can be clicked and expanded on a line, a circle is preferred over a line.

Chapter 7

Conclusions and Future Work

In my thesis, I showed that providing non-experts with readily available tools to mine and interpret frequent patterns is beneficial. PatternShow, a cloud-based scalable frequent pattern mining web interface with a frequent pattern visualization tool was implemented with a novel frequent pattern visualization, BundleVis, designed. A user study was carried out to evaluate the whole system and shows the effectiveness and efficiency of PatternShow.

7.1 Conclusions

In this MSc thesis, I designed, implemented, and evaluated PatternShow—which is a cloud-enabled web application, along with a novel frequent pattern visualization tool called BundleVis. PatternShow is scalable and readily available for non-expert miners to mine and easily interpret frequent patterns.

In the beginning of this thesis, I asked four research questions I would like to explore and answer. Having all the work done, I now provide an answer to each question.

In question one, I asked: ***Can we make a data mining system readily available for non-expert miners to mine frequent patterns?*** Chapter 5 gives the answer: PatternShow. It provides a web interface that allows non-expert users to register/log in to simply and securely upload data and get frequent patterns.

In question two, I asked: ***Can we make a visualization tool readily available for non-expert miners to better interpret mined frequent patterns?*** Chapter 4 gives the answer: BundleVis. It allows non-expert miners to visually interpret frequent patterns through interactive exploration. BundleVis is built into PatternShow to give users an integrated user experience.

The third question I asked is: ***Can we do both in a system that is scalable and non-expert miners do not have to worry about the computing infrastructure?*** The backend of PatternShow discussed in Chapter 5 is the answer. PatternShow leverages cloud computing and Apache Spark. It provides users, especially non-experts, with scalable frequent pattern mining. Non-expert miners do not need to worry about the computing resources because the frequent pattern mining algorithm is run distributed somewhere on the Internet.

In the last question, I asked: ***Can we make the system user-friendly in terms of effectiveness and efficiency?*** Chapter 6 gives the answer. In the user study I conducted on PatternShow, participants are asked to finish a set of tasks and answer a questionnaire. In the analysis based on task completion time and questionnaire answers, data shows the effectiveness and efficiency of PatternShow.

7.2 Future Work

My thesis, the PatternShow that I developed, is only the first effort to provide frequent pattern mining to non-expert data miners. Nothing is perfect at the outset and there is still space for improvement. It would be an ongoing effort in order to improve PatternShow progressively.

As people use smart phones more than desktop or laptop computers, it would be interesting to see a user study conducted on PatternShow with participants using smart phones. This helps to reach out even more non-expert users and makes PatternShow more user-friendly to non-expert users. The user study can be similar to what I did in Chapter 6: a set of tasks and a user satisfaction questionnaire. One can start the evaluation with the same tasks and questionnaire to make a comparison, and then iteratively perform more user studies when improvements are implemented. In terms of the user study, another future direction can be implementing the potential improvements discussed in Chapter 6 while analyzing evaluation results and conducting more user studies in order to iteratively refine PatternShow.

The backend of PatternShow can also be improved. One limitation of the current implementation is that the Apache Spark cluster has to be manually turned off, which is inefficient in terms of money when nobody is mining frequent patterns. The reason is that turning on the cluster takes 1 to 3 minutes, which is a long wait for users waiting to have data mined. There is currently no solution to this, which leads to an opportunity to do research.

Bibliography

[AIS93] R. Agrawal, T. Imielinski, and A. Swami, “Mining association rules between sets of items in large databases,” in *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data*, 1993, pp. 207–216.

[ALSG12] M. Amaro, N. C. Lino, C. A. Siebra, and A. Guedes, “The knowledge as a service metaphor and its use for building convergence environments,” in *Proceedings of the 18th Brazilian Symposium on Multimedia and the Web*, 2012, pp. 107–110.

[AS94] R. Agrawal and R. Srikant, “Fast algorithms for mining association rules,” in *Proceedings of the 20th International Conference on Very Large Data Bases*, 1994, pp. 487–499.

[BOH11] M. Bostock, V. Ogievetsky, and J. Heer, “D3; data-driven documents,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, pp. 2301–2309, 2011.

[BSH13] G. Bothorel, M. Serrurier, and C. Hurter, “Visualization of frequent itemsets with nested circular layout and bundling algorithm,” in *Proceedings* *of International Symposium on Visual Computing, Part II*, 2013, pp. 396–405.

[BWJL98] C. Bettini, X. S. Wang, S. Jajodia, and J.-L. Lin, “Discovering frequent event patterns with multiple granularities in time sequences,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 10, no. 2, pp. 222–237, 1998.

[CHL11] C. L. Carmichael, Y. Hayduk, and C.-S. Leung, “Visually contrast two collections of frequent patterns,” in *2011 IEEE 11th International Conference on Data Mining Workshops*, 2011, pp. 1128–1135.

[CL10] C. L. Carmichael and C. K.-S. Leung, “CloseViz: Visualizing useful patterns,” in *Proceedings of the ACM SIGKDD Workshop on Useful Patterns*, 2010, pp. 17–26.

[CPY98] M.-S. Chen, J. S. Park, and P. S. Yu, “Efficient data mining for path traversal patterns,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 10, no. 2, pp. 209–221, 1998.

[FPSM92] W. J. Frawley, G. Piatetsky-Shapiro, and C. J. Matheus, “Knowledge discovery in databases: An overview,” *AI magazine*, vol. 13, no. 3, p. 57, 1992.

[FZRL08] I. Foster, Y. Zhao, I. Raicu, and S. Lu, “Cloud computing and grid computing 360-degree compared,” in *Grid Computing Environments Workshop*, 2008, pp. 1–10.

[GMCE15] K. Grolinger, E. Mezghani, M. A. Capretz, and E. Exposito, "Collaborative knowledge as a service applied to the disaster management domain," *International Journal of Cloud Computing*, vol. 4, no. 1, pp. 5–27, 2015.

[GMF06] D. Guedes, W. Meira, and R. Ferreira, "Anteater: A service-oriented architecture for high-performance data mining," *IEEE Internet Computing*, vol. 10, no. 4, pp. 36–43, 2006.

[Gor08] D. Gorea, "Knowledge as a service. an online scoring engine architecture," in *Proceedings of the Third International Multi-Conference on Computing in the Global Information Technology*, 2008, pp. 1–6.

[GSPZ14] D. Garca-Saiz, C. Palazuelos, and M. Zorrilla, "Data mining and social network analysis in the educational field: An application for non-expert users," in *Educational Data Mining*, A. Pena-Ayala, Ed. Springer, 2014, pp. 411–439.

[HDY99] J. Han, G. Dong, and Y. Yin, "Efficient mining of partial periodic patterns in time series database," in *Proceedings of 15th International Conference on Data Engineering*, 1999, pp. 106–115.

[HPY00] J. Han, J. Pei, and Y. Yin, "Mining frequent patterns without candidate generation," in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, vol. 29, no. 2, 2000, pp. 1–12.

[JS91] B. Johnson and B. Shneiderman, "Tree-maps: A space-filling approach to the visualization of hierarchical information structures," in *Proceedings of 1991 IEEE Conference on Visualization*, 1991, pp. 284–291.

[LC09] C. K.-S. Leung and C. L. Carmichael, "FpViz: a visualizer for frequent pattern mining," in *Proceedings of the ACM SIGKDD Workshop on Visual Analytics and Knowledge Discovery: Integrating Automated Analysis with Interactive Exploration*, 2009, pp. 30–39.

[LC10] ——, "FpVAT: a visual analytic tool for supporting frequent pattern mining," *ACM SIGKDD Explorations*, vol. 11, no. 2, pp. 39–48, 2010.

[LIC08a] C. K.-S. Leung, P. P. Irani, and C. L. Carmichael, "FIsViz: A frequent itemset visualizer," in *Proceedings of the 12th Pacific-Asia Conference on Advances in Knowledge Siscovery and Data Mining*, 2008, pp. 644–652.

[LIC08b] C. K.-S. Leung, P. P. Irani, and C. Carmichael, "WiFIsViz: Effective visualization of frequent itemsets," in *Proceedings of the Eighth IEEE International Conference on Data Mining*, 2008, pp. 875–880.

[LJ12] C. K.-S. Leung and F. Jiang, "RadialViz: an orientation-free frequent pattern visualizer," in *Proceedings of the 16th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining-Volume Part II*, 2012, pp. 322–334.

[LJI11] C. K.-S. Leung, F. Jiang, and P. P. Irani, "FpMapViz: A space-filling visualization for frequent patterns," in *Proceedings of the 2011 IEEE 11th International Conference on Data Mining Workshops*, 2011, pp. 804–811.

[LLN00] L. V. Lakshmanan, C. K.-S. Leung, and R. T. Ng, "The segment support map: Scalable mining of frequent itemsets," *ACM SIGKDD Explorations*, vol. 2, pp. 21–27, 2000.

[LTC12] I. K. Lai, S. K. Tam, and M. F. Chan, "Knowledge cloud system for network collaboration: A case study in medical service industry in China," *Expert Systems with applications*, vol. 39, no. 15, pp. 12 205–12 212, 2012.

[LWZ+08] H. Li, Y. Wang, D. Zhang, M. Zhang, and E. Y. Chang, "Pfp: Parallel fp-growth for query recommendation," in *Proceedings of the 2008 ACM Conference on Recommender Systems*, 2008, pp. 107–114.

[MAG13] S. Moens, E. Aksehirli, and B. Goethals, "Frequent itemset mining for big data," in *Proceedings of the 2013 IEEE International Conference on Big Data*, 2013, pp. 111–118.

[MCS00] B. Mobasher, R. Cooley, and J. Srivastava, "Automatic personalization based on web usage mining," *Communications of the ACM*, vol. 43, no. 8, pp. 142–151, 2000.

[PCY95] J. S. Park, M.-S. Chen, and P. S. Yu, "An effective hash-based algorithm for mining association rules," in *Proceedings of the 1995 ACM SIGMOD International Conference on Management of Data*, 1995, pp. 175–186.

[PHL+01] J. Pei, J. Han, H. Lu, S. Nishio, S. Tang, and D. Yang, "H-mine: Hyper-structure mining of frequent patterns in large databases," in *Proceedings* *of the 2001 IEEE International Conference on Data Mining*, 2001, pp. 441–448.

[PW14] A. Perer and F. Wang, "Frequence: Interactive mining and visualization of temporal frequent event sequences," in *Proceedings of the 19th International Conference on Intelligent User Interfaces*, 2014, pp. 153–162.

[RS14] P. Ramya and S. Sasirekha, "Text mining system for non-expert miners," *International Journal of Computer Science and Information Security*, vol. 12, no. 5, pp. 14–18, 2014.

[SN00] S. Sarawagi and S. H. Nagaralu, "Data mining models as services on the internet," *ACM SIGKDD Explorations*, vol. 2, no. 1, pp. 24–28, 2000.

[SSW10] K. Stanoevska-Slabeva and T. Wozniak, "Cloud basics an introduction to cloud computing," in *Grid and Cloud Computing*, 2010, pp. 47–61.

[TVS01] A. S. Tanenbaum and M. Van Steen, *Distributed systems: principles and paradigms*. Prentice Hall PTR, 2001.

[WXL99] K. Wang, C. Xu, and B. Liu, "Clustering transactions using large items," in *Proceedings of the Eighth International Conference on Information and Knowledge Management*, 1999, pp. 483–490.

[XZ05] S. Xu and W. Zhang, "Knowledge as a service and knowledge breaching," in *Proceedings of 2005 IEEE International Conference on Services Computing*, vol. 1, 2005, pp. 87–94.

[ZGS13] M. Zorrilla and D. Garca-Saiz, “A service oriented architecture to provide data mining services for non-expert data miners,” *Decision Support Systems*, vol. 55, no. 1, pp. 399–411, 2013.

[ZL01] O. R. Zaiane and J. Luo, “Web usage mining for a better web-based learning environment,” in *Proceedings of Conference on Advanced Technology for Education*, 2001, pp. 60–64.

[ZZB11] F. Zhang, Y. Zhang, and J. Bakos, “Gpapriori: Gpu-accelerated frequent itemset mining,” in *Proceedings of the 2011 IEEE International Conference on Cluster Computing*, 2011, pp. 590–594.

Appendix A

Course Dataset

A.1 ID-Description Mapping

208	Analysis of Algorithms
215	Object Orientation
228	Introduction to Computer Systems
301	Distributed Computing
304	Technical Communication in Computer Science
317	Analysis of Algorithms and Data Structures
335	Software Engineering 1
343	Operating Systems
362	Professional Practice in Computer Science
402	Human-Computer Interaction 2
405	Project Management
406	Topics in Computer Science
435	Software Engineering 2
436	Machine Learning
438	Database Implementation
442	Advanced Design and Analysis of Algorithms
443	Operating Systems 2
452	Undergraduate Honours Project
455	Real-Time Systems
456	Industrial Project
458	Computer Security
472	Computer Networks 2

A.2 The Dataset

```
208 215 343 438
438 452
304 406 438 458
362 435 438 458 472
301 362 438
317 335 343 438
304 402 405 438
402 435 438
208 215 343 438
317 402 435 438
438
301 304 343 438
438
228 438
301 362 435 438
304 343 362 435 438
438 443 455 458 472
402 438 458 472
317 438 458
435 438 455 458
438 443 452 456 458
438
208 215 438
406 435 438 458
435 438 458
208 215 228 438
402 435 438 458
438 455 472
435 438 455 458
435 438
435 438
317 335 406 438 455 458
304 438 455
304 406 436 438 458
435 438
438 458
335 438
435 438 472
438 456 472
438 455 458
```

```
438 455
304 438 455 458
435 438 456
304 438
435 438 442 455 458
438 456 472
362 402 435 438 456
301 435 438
362 438
438
228 317 335 438
317 335 343 438
```

Appendix B

Evaluation Documents

I presented four documents to participants during the evaluation of PatternShow. They are informed consent form, a questionnaire asking if the participant is a data mining expert, tasks, and a user satisfaction questionnaire.

UNIVERSITY OF MANITOBA

DEPARTMENT OF COMPUTER SCIENCE

Winnipeg, Manitoba
Canada R3T 2N2
(204) 474-8313
FAX (204) 474-7609

Informed Consent Form

This consent form is only part of the process of informed consent. It should give you a basic idea of what the research is about and what your participation will involve. We recommend you keep a copy this form for your records and reference. If you would like more information or details about this research, please feel free to contact Mr. Zhao Han (umhan35@myumanitoba.ca). Please take the time to read this carefully.

The purpose of this user study is to evaluate the web interface called PatternShow, which is designed to allow non-expert users to upload some data to be mined and understand frequent patterns through visualization. A frequent pattern can be a shopper basket of fruits and vegetables that frequently bought in a grocery store. If the store manager knows this information, staff in the store can then place those frequently bought merchandise items in proximity, which both brings convenience to customers.

Here, in this user study, you are invited to use PatternShow, perform some tasks, and answer a questionnaire about the quality of PatternShow. As we are studying how we can improve PatternShow to make navigation of a frequent pattern mining process easier—especially for non-experts, a screencast (i.e., a digital recording of the computer screen capturing user input & computer output) will be taken without any sound records. For analytical purposes, we will keep track of both the completion time of each task and the accuracy of the answer. However, you should not feel rushed.

Participation in this study is voluntary, and will take no more than 30 minutes of your time. No personal information will be collected for this study. Data collected for this study will be retained for a period of maximum five years in a locked office in the EITC building, University of Manitoba, to which only researchers associated with this study have access and will only be used to evaluate the performance of the recommendation system. We anticipate that results will be made available at our lab website (http://dblab.cs.umanitoba.ca/) and published within 1 to 2 years. If you are interested in the results, please email the researchers at the email given. Again, no personal information about your involvement will be collected.

By signing this form, you indicate that you understand to your satisfaction the information regarding participation in the research project and agree to participate as a subject. By doing this, you also confirm that you are of the age of majority in Canada (18 years or more). In no way this waive your legal rights nor release the researchers, sponsors, or involved institutions from their legal and professional responsibilities. You are free to withdraw from the study simply by closing your browser or navigating away from the survey website at any time without prejudice or consequence; in this case, the incomplete data will not be saved. If you prefer not to answer any question, you may simply omit it.

The University of Manitoba may look at your research records to see that the research is being done in a safe and proper way.

This research has been approved by the University of Manitoba Joint-Faculty Research Ethics Board. If you have any concerns or complaints about this project, please contact Dr. Carson Leung (Carson.Leung@umanitoba.ca) or the Human Ethics Coordinator/Secretariat at +1 (204) 474-7122.

Researcher's Signature: ____________________ Date: ________________

Participant's Signature: ____________________ Date: ________________

UNIVERSITY OF MANITOBA

DEPARTMENT OF COMPUTER SCIENCE

Winnipeg, Manitoba
Canada R3T 2N2
(204) 474-8313
FAX (204) 474-7609

Participant: Are you a data mining expert or not?

1. Do you consider yourself as a data mining expert or non-expert?

   □ Data mining expert □ Data mining non-expert

2. Do you know what data mining is?

   □ No □ Yes, please explain:

3. Do you know what frequent pattern mining is?

   □ No □ Yes, please explain:

UNIVERSITY
OF MANITOBA

DEPARTMENT OF COMPUTER SCIENCE

Winnipeg, Manitoba
Canada R3T 2N2
(204) 474-8313
FAX (204) 474-7609

Tasks

Please feel free to work on the tasks at your own pace, and have a break between them. You are free to skip the task or leave if you do not feel comfortable with any task.

Imagine you are a store manager, you have an anonymous records of 88,163 transactions and are saved in a data file `retail.txt` on desktop.

Now please use the website to explore the data and try to complete the following tasks:

1. Please upload the data file `retail.txt` located on desktop.
2. As stated before: there are 88,163 transactions, please fill up the minimum support field (In this study, we assume if there are 8 transactions containing an item, then the item is frequent).
3. Which are the most popular merchandise items customers usually bought with item 1198 (apple)?
4. How many customers bought both item 1198 (apple) and item 48 (banana)?
5. How many customers bought item 1198 (apple), item 48 (banana) together with item 39 (cherry)?
6. Among the customers who bought item 1198 (apple), how many other customers bought both item 48 (banana) and item 39 (cherry)?
7. What other information did you observe from the visualization?

UNIVERSITY OF MANITOBA

DEPARTMENT OF COMPUTER SCIENCE

Winnipeg, Manitoba
Canada R3T 2N2
(204) 474-8313
FAX (204) 474-7609

User Satisfaction Questionnaire

This questionnaire is the last part of this study:

1. The help note for “minimum support” is helpful to understand the concept.

   Strongly disagree □ Disagree □ Ok □ Agree □ Strongly agree □

2. I find it useful to specify the item of interest.

   Strongly disagree □ Disagree □ Ok □ Agree □ Strongly agree □

3. It is easy to find the most popular merchandise items.

   Strongly disagree □ Disagree □ Ok □ Agree □ Strongly agree □

4. It is easy to find how many customers bought both apples and bananas.

   Strongly disagree □ Disagree □ Ok □ Agree □ Strongly agree □

5. It is easy to find how many customers bought apples (item 10258), bananas (item 78963) together with cherries (item 15987).

   Strongly disagree □ Disagree □ Ok □ Agree □ Strongly agree □

6. It is easy to find that, among the customers who bought apples (item 10258), how many other customers bought both bananas (item 78963) and cherries (item 15987).

   Strongly disagree □ Disagree □ Ok □ Agree □ Strongly agree □

7. Do you have any other comments about the software that were presented to you (e.g., level of usefulness)?

Thank you for participating in this study, your input is truly appreciated. Results of this study will be made available at http://dblab.cs.umanitoba.ca/ and are expected to be published within 1 to 2 years. If you have any concern after this study, please contact Dr. Carson Leung (Carson.Leung@umanitoba.ca) or Mr. Zhao Han (umhan35@myumanitoba.ca).